
# NAVAL POSTGRADUATE SCHOOL

**MONTEREY, CALIFORNIA**

# THESIS

**EVOLUTIONARY CAPABILITY DELIVERY OF COAST GUARD MANPOWER SYSTEM**

by

Heather D. Skowron

June 2014

Thesis Advisor: Ronald E. Giachetti
Co-Advisor J. Marc Aparicio

**Approved for public release; distribution is unlimited**

THIS PAGE INTENTIONALLY LEFT BLANK

# REPORT DOCUMENTATION PAGE

*Form Approved OMB No. 0704-0188*



| **1. AGENCY USE ONLY** *(Leave blank)* | **2. REPORT DATE** June 2014 | **3. REPORT TYPE AND DATES COVERED** Master's Thesis |
|---|---|---|

**4. TITLE AND SUBTITLE**
EVOLUTIONARY CAPABILITY DELIVERY OF COAST GUARD MANPOWER SYSTEM

**5. FUNDING NUMBERS**

**6. AUTHOR(S)** Skowron, Heather D.

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**
Naval Postgraduate School
Monterey, CA 93943-5000

**8. PERFORMING ORGANIZATION REPORT NUMBER**

**9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES)**
N/A

**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**

**11. SUPPLEMENTARY NOTES** The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol number ____N/A____.

**12a. DISTRIBUTION / AVAILABILITY STATEMENT**
Approved for public release; distribution is unlimited

**12b. DISTRIBUTION CODE**

**13. ABSTRACT (maximum 200 words)**

The United States Coast Guard (CG) is currently using Microsoft© products as the main means of completing Manpower Requirement Determinations (MRDs). This does not allow the CG to produce reliable, repeatable, and defendable determinations nor does it assist in meeting the 5-year evaluation cycle of 128 unit types. For the CG to ensure human capital decisions are scientifically based and become more efficient in MRD completion, the CG will transition to a MRD system that automates much of the process. To develop the system, this thesis argues and recommends the use of an Iterative Incremental Development (IID) process model to deliver capability in increments driven by schedule. The model consists of seven iterations with a six-month duration to complete all iteration activities. In addition, this thesis proposes a three-tier system's architecture to act as a blueprint of the target system. The three-tier architecture consists of a data, application, and presentation layer. The results of this thesis is a process model and systems architecture that will guide the CG's development team through development and acquisition of the system while enabling user feedback throughout the process and reducing risk.

| **14. SUBJECT TERMS** United States Coast Guard, Systems Engineering, Manpower, Iterative Incremental Development, Model Based System Engineering, system architecture, Manpower Requirements Determination | **15. NUMBER OF PAGES** 109 |
|---|---|
| | **16. PRICE CODE** |

| **17. SECURITY CLASSIFICATION OF REPORT** | **18. SECURITY CLASSIFICATION OF THIS PAGE** | **19. SECURITY CLASSIFICATION OF ABSTRACT** | **20. LIMITATION OF ABSTRACT** |
|---|---|---|---|
| Unclassified | Unclassified | Unclassified | UU |

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

**Approved for public release; distribution is unlimited**

# EVOLUTIONARY CAPABILITY DELIVERY OF COAST GUARD MANPOWER SYSTEM

Heather D. Skowron
Lieutenant Commander, United States Coast Guard
B.A., Rivier University, 1999

Submitted in partial fulfillment of the
requirements for the degree of

**MASTER OF SCIENCE IN SYSTEMS ENGINEERING**

from the

**NAVAL POSTGRADUATE SCHOOL**
**June 2014**

Author: Heather D. Skowron

Approved by: Ronald E. Giachetti
Thesis Advisor

J. Marc Aparicio
Co-Advisor

Clifford Whitcomb
Chair, Department of Systems Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

# ABSTRACT

The United States Coast Guard (CG) is currently using Microsoft products as the main means of completing Manpower Requirement Determinations (MRDs). This does not allow the CG to produce reliable, repeatable, and defendable determinations nor does it assist in meeting the five-year evaluation cycle of 128 unit types. For the CG to ensure human capital decisions are scientifically based and become more efficient in MRD completion, the CG will transition to a MRD system that automates much of the process. To develop the system, this thesis proposes the use of an Iterative Incremental Development (IID) process model to deliver capability in increments driven by schedule. The model consists of seven iterations with a six-month duration to complete all iteration activities. In addition, this thesis proposes a three-tier system's architecture to act as a blueprint of the target system. The three-tier architecture consists of a data, application, and presentation layer. The results of this thesis is a process model and systems architecture that will guide the CG's development team through development and acquisition of the system while enabling user feedback throughout the process and reducing risk.

THIS PAGE INTENTIONALLY LEFT BLANK

# TABLE OF CONTENTS

# LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

# LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

# LIST OF ACRONYMS AND ABBREVIATIONS

| | |
|---|---|
| ACS | automated case support |
| AIS | Automated Information System |
| ALMIS | Aviation Logistics Management Information System |
| AOPS | Abstract of Operations System |
| AOR | area of responsibility |
| CG | Coast Guard |
| CG-1B | Human Resources Strategy & Capability Development Office |
| CGBI | Coast Guard Business Intelligence System |
| CMPlus | Configuration Management Plus |
| CV-1 | capability vision |
| CV-3 | capability phasing |
| FBI | Federal Bureau of Investigation |
| FLS | Fleet Logistics System |
| GAO | Government Accountability Office |
| IID | iterative incremental development model |
| IT | information technology |
| MA | major accomplishment |
| MRA | manpower requirements analysis |
| MRD | manpower requirement determination |
| OV-1 | Operational View |
| POE | projected operational environment |
| PV-2 | project timeline |
| ROC | required operational capability |
| SDLC | system development life cycle |
| SE | systems engineering |
| SEI | Software Engineering Institute |
| SV-1 | systems interface description |
| SV-4 | systems functionality description |
| SV-5b | operational activity to systems traceability matrix |
| SV-9 | systems technology & skills forecast |

| | |
|---|---|
| TMT | Training Management Tool |
| USCG | United States Coast Guard |
| VCF | Virtual Case File |

# EXECUTIVE SUMMARY

The work contained in this thesis proposes and argues for an Iterative Incremental Development (IID) process model and three-tier system architecture to aid the Coast Guard (CG) in the development of a Manpower Requirements Determination (MRD) system. The successful acquisition of Information Technology (IT) is a challenge for organizations looking to automate business processes. To address the challenges and ensure development is successful, especially in the current fiscally limited budgets, the CG will need to ensure that development is low risk.

The CG uses Manpower Requirements Analysis (MRAs) to collect the necessary manpower data to determine the human capital needs of CG units (U.S. Coast Guard [USCG] n.d.). The analysis is then used to complete a MRD, which determines what human capital meets the needs. These processes are currently both completed with software support from Microsoft Excel. The CG needs a system that has more robust capability than Microsoft Excel that efficiently and accurately completes the analysis and modeling needed for high quality MRDs (U.S. Coast Guard 2008). In addition, the CG has a target completion rate of 26 MRDs annually or 128 MRDs every five-year cycle. Current tools do not automate the process to a level that produces MRDs with the speed needed to meet this goal. A new MRD system will also provide accuracy and scientific rigidity to the process due to the inherent ability of software to manage data and perform calculations. Software will also allow for the production of models and graphics that will enhance analysts' ability to make manpower decisions. A new MRD system will reduce the MRD evolution time, create a repeatable process, and ensure MRDs are based on dependable and reliable data and analysis.

Technology's fast paced growth and complexity make IT system development extremely challenging. Requirements creep, testing and evaluation issues, lack of user participation in development, and inaccurate risk analysis all add to the challenge. To mitigate these challenges, system engineering work is critical. Model-based system engineering and systems architectures provide a means of developing a system through visual representation of the process and components, one that is simpler, reduces

complexity, ensures all member of the development team understand the system and process, and provides a means of traceability for function and capability delivery (Giachetti 2010).

Four different process models were considered for the MRD system including the waterfall, "Vee," spiral, and IID. The IID model was the process model selected due to the models incremental approach to developing a system, the accommodation of up-front systems engineering, and the integrated paths of feedback. The IID process model for the MRD system includes seven iterations, with each iteration containing five activities: analysis, build, implement, deploy, and support. The IID process model for the MRD system has an 18-month evolution from start to finish with each iteration occurring over a 4-6 month time frame. Each iteration within the model overlaps to keep the overall schedule short and the project consistently moving forward. With the IID process model, this work presents a three-tier systems architecture for the MRD system. The architecture consists of the components needed to produce the MRD system and has three-tiers, or layers, including a data, application and presentation layer. The MRD architecture is designed to evolve in a consistent progression, in step, with the IID process models evolution. The architecture provides the CG with a means of meeting architectural policies and requirements while providing the development team with a blueprint of the system that ensures a better understanding of the system and the integration of its components.

The IID process model and three-tier architecture, in conjunction with associated up-front systems engineering work, provides the CG with the best means to develop the MRD system. The development team will benefit in having an organized systematic method to develop the system and ensure the product delivered meets the function and capability needs of the CG. The process model will reduce the risk, technical complexity, and other challenges mentioned herein to ensure that CG funding is not wasted. In addition, the process model and systems architecture will also allow the CG to better develop the specification requirements for acquisition and ensure the product delivered is usable by CG personnel; which has been a problem in previous attempts at a system.

## LIST OF REFERENCES


Giachetti, Ronald. 2010. Design of Enterprise Systems Course. Monterey, CA.

U.S. Coast Guard. (n.d.*). Coast Guard Staffing Logic and Manpower Requirements Manual, Vol II (COMDTINST M5310.5)*, Washington, D.C.

U.S. Coast Guard. 2008. "Manpower Requirements Determination (MRD) Enterprise Development Team Charter (Update 1). Accessed June 18, 2014. http://www.uscg.mil/ff21/docs/MRD_team_charter_updated.pdf.

THIS PAGE INTENTIONALLY LEFT BLANK

# ACKNOWLEDGMENTS

I would like to thank Commander J. Marc Aparicio for acting as my thesis co-advisor and facilitating the idea of completing this work for the Coast Guard through this thesis. I would like to sincerely thank my thesis advisor, Professor Ronald Giachetti, for his tireless support and guidance while developing the work within this thesis. In addition, I thank Commander Patricia Kutch for providing the U.S. Coast Guard research data needed to complete this work. I would like to further thank Ms. Barbara Berlitz, a technical writing instructor in the Systems Engineering Department, and the staff at the Thesis Processing Office, at the Naval Postgraduate School, for their editing and revision recommendations. Lastly, I would like to extend my gratitude to every professor within the Systems Engineering Department; their dedication and ability to teach the concepts needed to complete systems engineering work were invaluable to me when completing this thesis.

THIS PAGE INTENTIONALLY LEFT BLANK

# I. INTRODUCTION

## A. MOTIVATION

The successful acquisition of Information Technology (IT) systems presents an ever-growing challenge for organizations looking to automate work processes. The quick pace of software evolution, the lack of IT development knowledge within project teams and developers, and the length of time it takes to acquire the developed system from start to finish, all contribute to this challenge. To ensure project teams acquiring IT systems have the opportunity for success throughout development and during subsequent acquisitions, the selection of a process model from which to develop and architect the system should be carefully considered. This thesis provides a suitable process model and system architecture for the United States Coast Guard (USCG) to develop and acquire a Coast Guard (CG) Manpower Requirements Determination (MRD) system. This research found that the best process model for the CG and this system is the Iterative Incremental Development (IID) process model. The IID process model addresses many challenges associated with IT system acquisition. The process model can lower risk, shorten processing time during development, allow for integration of feedback as the system is delivered, which will ensure the system meets the usability needs of users.

The CG uses two business processes to manage human capital: Manpower Requirements Analysis (MRA) and Manpower Requirements Determination (MRD). Analysts conduct MRA's in order to analyze and determine the manpower needs of each unit type within the CG. The MRD is then conducted, using the MRA, to determine how best to meet the determined needs. The MRA consists of collecting mission requirements data, and then assessing that data against allowances, standards, regulations, and policies, to create models of manpower requirements needed for each unit type within the CG. The data collection and analysis performed during the MRA is currently conducted with software support from Microsoft Excel. Microsoft Excel is able to store data and create basic templates for reports and data but is not an automated means of completing the analysis. Microsoft Excel lacks a robust program capability for completing the type of analysis and modeling a MRD system requires. Upon completion of the MRA, analysts

begin the MRD, which is completed by analysts in the same manner as the MRA. MRDs are subsequently completed with little scientific rigidity and optimization of determinations. From this point forward in this report, the MRD will be assumed to include the MRA except where noted.

The CG's goal is to complete 26 MRDs annually or 128 MRDs every five-year cycle. Current methods of MRD completion explained only enable approximately 1 MRD every seven to twelve months. The methods used to complete MRDs are directly related with the speed with which CG analysts complete MRAs and MRDs. Therefore the method is not optimal, with a typical MRD taking up to a year from start to finish. In addition to the process being slow, no scientific methodology or modeling is used to consistently repeat the process. Therefore, the process is not defendable when being used to justify manpower needs for the service. Mangers within the CG "do not have confidence that any particular set of human capital requirements is based on industrial engineering principles, or any objective science, and cannot compare sets of requirements to optimize human resource allocation" (United States Coast Guard [USCG] 2006). The CG has noted, in the Commandant's Intent Action Order 8, that "staffing standards are obsolete" and that "no process exists to remedy this state" (USCG 2006). To remedy the method deficiencies, the system needs to transition to a semi-automated process, delivered by an IT system. The use of an IT system, with MRD specific software, will ensure data are readily available, up to date, and organized for use. Software will also provide accuracy and speed to formula calculations, produce models and graphics quickly, and produce reports with ease. Automating the system will give analysts the ability to consistently apply the same scientific methods to each CG unit when completing MRDs. The overall automation the system will reduce time, create a repeatable process, and ensure MRDs are defendable based on the data and analysis methods used.

While presenting the solution as an IT system, it is important to note the alternatives to this. There are three possible alternatives, hire more contractor support, contract with the Navy for more support than already received, or hire more CG analysts. Each of these alternatives is a personnel-hiring decision that has associated costs. Hiring

additional personnel is unrealistic in the current budget climate and is costly means of improving current business methods. Hiring personnel is costly in both the short term and long term, as one has to consider not only the individual's salary but also the cost of training and benefits such as retirement, medical insurance, and others. An IT system is not without its associated costs, including training and upgrade costs. Regardless of the hiring of personnel to speed up the number of MRDs completed within a five-year cycle, additional personnel will not improve the data or analysis of the data, which are main components of the process. Overall, hiring additional personnel will not remedy the lack of consistency, repeatability, or scientific rigidity needed to complete an MRD. The only real solution for the CG when analyzing human capital is a system that automates much of the MRD process.

The MRD process presents itself as an ideal candidate for the IID process model. It provides a solution near term and long term and the CG cannot afford to wait long periods of time if they desire a remedy their current manpower determination process method. The IID process model will allow some capability to be delivered early in development allowing the CG to develop subsequent capability and functionality in increments. The MRD process is built on HR business practices, namely data collection, data analysis and data output. These business practices embedded in the process of MRD completion make the MRD buildable in increments in that each layer of the process can be automated at different times. Unlike building a typical ship or small boat, an IT system can be developed in pieces and stand-alone components that can provide some level of functionality and capability as it is built. Computer software and hardware inherently allow for integration of new components at later times. The ability to deliver capability early, in combination with the inherent nature of IT systems being buildable over time, makes the IID process model ideal. This report will continue to show the selection of the IID Model as the most optimal process model for the CG MRD system and place emphasis on its ability to incorporate feedback from users.

The IID process model will only work with an appropriate systems architecture that assists in "achieving a well-defined system in both operational and physical domains" (Whitcomb n.d.). The system architecture provides a visual blueprint of the

system components used to build the system and demonstrates the relationship between the components (Scheinholtz 1998). The architecture proposed within this thesis is a three-tier architecture, for the MRD system, consists of a data, application, and presentation layer (Giachetti 2010). The architecture limits the complexity of the system by using common hardware and the existing CG network while still allowing for the development of new software for a database and MRD system application. The final system architecture will be broken into workable smaller pieces that all for the system to evolve as the IID process model progresses through each of the iterations. The MRD system architecture will make the system easy to understand, assist in development, and ensure the development team meets CG architecture requirements.

## B. CURRENT BUSINESS PRACTICE WITHIN CG-1B4

CG-1B4 is currently staffed with two policy team members and seven analysts. CG-1B4 also receives support from the Navy Manpower Analysis Center (NAVMAC) in the form of five analysts and contractor assistance. CG-1B4's current as-is MRD system uses multiple Microsoft Excel databases and template generators. The mix of analysts and as-is system only does not enable the CG to meet a 26 per year MRD completion rate.

The as-is system requires manual data input into Microsoft Excel templates for each MRD. This requires sifting through all CG personnel requirements in service manuals, directives, instructions, policies, unit directives and instructions pertinent to the unit the MRD is being conducted on, called the MRD Sponsor. This method can lead to a failure to collect all pertinent data and possible information gaps. Once this is completed the analysts are required to engage the MRD Sponsor. The sponsor has a large contribution to the MRD as they must take what data the analyst has compiled, check it for accuracy, and ensure all pertinent reference requirements are met. This is very time consuming both for the analyst and the MRD Sponsor, which only results in building a snapshot of manpower numbers and skills required by the unit. The analyst must then allocate personnel to the manpower numbers and skill categories by hand using both personal knowledge and experience to find the optimal mix of personnel to meet unit

requirements. This creates a biased analysis; the analyst has a lot of room to enter their beliefs, feelings, and past experiences into what they determine. Each analyst is therefore not completing MRDs with the exact same knowledge. A software system will remove the bias by using consistent logic and formulas to determine the results of calculations and modeling, without feelings, beliefs, and past experience. Removing the bias, which can only be accomplished through automation of much of the MRD process, in conjunction with a reliable and accurate database, will ensure the MRD is defendable and repeatable.

## C. PROJECT BACKGROUND

The CG has made previous attempts to develop a MRD system. The last attempt to accomplish this was in 2008 and focused on delivering the capability to evaluate workload, calculate human resources and labor (CG-1B4 2008). Having reviewed the Project Management Plan from this attempt, the CG was delivered a database-based system with a user interface that allowed for user interaction. This database was designed by Dell Perot Systems and has the ability to perform the following functions:

- Manage Data: save data, add new records, upload of files, export data to Microsoft Excel
- Read Data: query and retrieve data
- Update Data: change existing data
- Delete Data: remove data from the database

The system delivered to the CG had a major flaw in that it was not usable by analysts because it contained no logical workflow processes to get to the output needed from the system. The system built contained 231 functional requirements that never addressed the functional need of users, which is to complete MRDs with automation and little input from analysts (U.S. Coast Guard 2008). The system was mainly a database-based system that allowed for some input and output of data but contained no logical workflow for the user to follow, no means of data and modeling output, and required a significant learning curve for data entry and use.

The MRD system that was delivered was also built with six non-functional requirements including availability, integrity, security, interoperability, usability, and performance. The non-functional requirement, as per the MRD system Functional Requirements document, version 2.3, states that the system shall:

1. Shall contain a standard and consistent user interface.
2. Shall allow users to employ "short-cuts" to execute system functions.
3. Shall allow users to reference common and standard data from a list of items. (U.S. Coast Guard 2008).

These three non-functional usability requirements do not address the environment of the system and user, the true workflow involved in the MRD and what aspects need human prompting or participation, the ease of learning and using the provided system, or whether the system is error tolerant. Development of this non-functional requirement would have helped guide the developer when designing how the functional delivery was to occur. While this would have helped, it would not have fully remedied the usability problems with the delivered system. The system was built from start to finish with little incorporated feedback, due to the lack of a process model with an organized means and timing for feedback incorporation. It is highly recommended that this pre-existing attempt to develop a system not be used.

## D. CG-1B4'S GOALS & OBJECTIVES

CG-1B4's goal is to obtain a system that allows for a definable, verifiable, and repeatable means of accomplishing the analysis that feeds the MRDs. The business objectives are as follows:

- The system must allow for the successful collection, measurement, and analysis of the human capital needed by each unit type within the CG.
- The system shall reduce the MRD process cycle time and labor hours required in completing each MRD.
- The system shall allow for a standard process of determining manpower requirements.

- The system shall increase the quality of human capital decisions and improve the visibility of human capital needs for the organization.

## E. IMPACT OF MRD SYSTEM

In order to ensure the largest impact possible from a new system, as stated in Chapter one section A of this report, this thesis argues a IID process model and three-tier system architecture that delivers capability in an incremental format to reduce both risk and cost to the CG. As portions of the system are delivered, analysts will be able to use each portion to improve their work production. This will provide for reduced man-hours in accomplishing MRDs at each phase. The system architecture will provide the blueprint within each iteration of the process model. Additionally, as the final phase is delivered, and full system capability is realized by CG-1B4, MRD production should meet, and possibly exceed, CG-1B4's business requirement of 26 MRDs per year.

In addition to those benefits stated above, the system will also benefit the quality of MRDs. With a system that has the capability to analyze and optimize workforce allocation, the quality of the MRD will increase. Having quantitative and optimized MRDs will not only benefit analysts within CG-1B4, but they will also benefit other CG organizational leadership. CG managers must routinely make manpower estimates when completing Requests for Proposals (RFPs), and when composing other budget documents. Having MRDs that are more accurate, rigorous, and quantifiable will ensure that these proposals and budget documents accurately reflect the needs, and or gaps, of the CG to both DHS and Congress. Lastly, CG managers will have better access to manpower data in order to develop more accurate mission accomplishment predictions and better identify shortages in personnel types and training. In summary, the system architecture provided herein, and the IT system upon incremental delivery, provides the following:

- Lower cost and risk to the CG.
- Quantifiable, rigorous, accurate manpower analysis.
- A reduction of man-hours required to complete MRDs.
- An increase in the number of MRDs produced by CG-1B4, yearly.

- Workforce data that accurately reflect the needs and gaps of the CG for use by both DHS and Congress.

## F. SCOPE OF THESIS

The scope of this thesis is to provide the CG with a process model and system architecture in order to develop and acquire a MRD system. Additional items included within the scope of this project are:

- A layout of each phase and the capability delivered with that phase of the system.
- Transitional use of the system as each phase is delivered.
- A schedule showing the delivery of each phase.
- Mapping of the system functions to capability.
- Mapping of the capability to system components.
- Risk analysis associated with each phase.

Things outside the scope of this thesis include the resources needed to carry out the acquisition of this system, the budget layout for acquiring this system, and specific identification of the Acquisition Decision Events (ADEs) on the timeline.

# II. IT ACQUISITION CHALLENGES

## A. INTRODUCTION

As technology grows and increases in complexity, so do the challenges faced by an organization trying to acquire any form of it. The growth in challenges is a direct result of the "increasingly critical role" that software now plays in all aspects of business practices (Creel and Ellison 2007). The challenges associated with IT systems include technical complexity, application of systems engineering methods, requirement creep, testing and evaluation issues, lack of user participation in development, and inaccurate risk analysis. These challenges are compounded by the imposed regulations in government organizations on system development and acquisition.

IT system projects that have failed to address these challenges usually result in cost and schedule overruns. In some instances it has caused cancellations of projects. A study conducted by the McKinsey & Company, along with the University of Oxford revealed that "on average, large IT projects run 45 percent over budget and 7 percent over time, while delivering 56 percent less value than predicted" (Block, Blumberg, and Lartz 2012). It was also found that the likelihood of a project to run over budget and schedule was greater the longer a project was scheduled for (Block, Blumberg, and Lartz 2012). To mitigate these challenges, the development process and systems architecture needs to be governed, namely by a process model best fitted to the project type. In an attempt to successfully architect a software solution for CG-1B, it is prudent to review these challenges and how these challenges affected the CG's first attempts to develop a system.

## B. PROJECT CHALLENGES

### 1. User Participation in Development

User participation in the development of an IT system can contribute to developers and project managers understanding the true needs of the users, yet so often this is overlooked. According to a report entitled "Chaos" by the Standish Group, the number one reason projects are "challenged," and number two reason for a "failed" project, is because of the "lack of user input" (Standish Group 1995). To clarify, user

input does not necessarily always mean to ‘ask the user’ rather it means that data should be gathered from watching users and gaining an understanding of the users job and business practice, in addition to direct input from users. This allows a developer to balance what people say they do with what they actually do (Nielsen 2001).

Functionality and usability are the two areas that concern the user the most. The functionality of the system is a measure of how well a system meets the output needs of a user. If the system does not perform a task that the user needs, the system will not be used. Having this understanding when developing a system aids in validating system functions, requirements, and capabilities against what are truly needed from the system. Developers must ensure the system created does not lack usability. This did not occur in the CG’s previous attempt to develop a MRD system. As discussed in Chapter I, Section C, the CG was delivered a highly unusable system, containing input and outputs and no sequential accomplishment of the business process associated with a MRD. Jakob Nielsen a usability expert states that usability includes five quality components: learnability, efficiency, memorability, errors, and satisfaction (Nielsen 2001). Further definition of these components is the ease of accomplishing tasks, speed at which tasks can be performed, ease of regaining proficiency with the system after a period of no use, the difficulty of recovering from errors made while using the system, and is the design pleasing to the user (Nielsen 2001). If a system is slow, difficult to use, or requires a long period of refresh training after an absence in use, a user will be less likely to use a system. These attributes can be addressed much easier when the “decisions of software developers were more likely to match the needs of the users” when the users where involved (Rowley 1996).

For a project to be successful, it must not only have good methods of communication established with the users for aiding in business practice understanding, but project managers must also use management protocols to ensure user involvement does not hamper the project (Standish Group 2009). Without user input the design of a product may technically meet the requirements, functions, and objectives of a system but may not be ‘usable’ by those it was created for. This was the most significant challenge the CG faced in the previous attempt to develop a MRD system. The functions delivered

were not usable by analysts. The system was mainly a database-based system with no logical workflow for the user to follow, no means of data output, and required a significant learning curve for data entry and use.

### 2. Technical Complexity

The definition of software, according to the Oxford English Dictionary, is "the programs and procedures required to enable a computer to perform a specific task, as opposed to the physical components of the system" (Oxford English Dictionary [OED] 2014). Hardware is what delivers or provides the mechanism for software to perform its function. Software performs the functions, tasks, and operations an individual needs to perform. Software is made up of many lines of code that are complex to the average software user. Due to the numerous lines of code within software, it is prone to bugs, or errors within. According to the Software Engineering Institute (SEI), for every 1,000 lines of code, there are approximately 10 defects, which results in 5,000-10,000 defects for a million lines of code (Gallagher 2008). To give a reference point for what would contain a million lines of code: the release of Windows NT 3.1 in 1993 contained over 5.6 million lines of code and had 30,000 errors in code that had to be fixed during the last year of development (*Wikipedia* 2014c). Software users and those managers that are part of IT project teams are usually unfamiliar with software and therefore unfamiliar with what is needed for testing and remedying these coding errors. The lack of understanding associated with software complexity comes with schedules that are unrealistic because of the failure to account for the time needed to ensure the software functions properly.

The complexity of software cannot be stressed enough and project team members need to ensure a level of understanding of software's inherent complexity in order to successfully manage an IT project from start to finish. Software according to Frederick Brooks is a "construct of interlocking concepts: data sets, relationships among data items, algorithms, and invocations of functions" and as such when the size increases "an increase in the number of different elements results" (Brooks 1987, 11). A lack of understanding of software by the project team can directly lead to inaccurate or poor requirement definition, inability to understand measures of performance and inability to

evaluate a system through testing and delivery. It has been noted that the knowledge and skills of team members managing systems where software is a significant component is insufficient (Jones 2011).

### 3. Changes to the Systems Engineering

Systems are created through a process called Systems Engineering (SE), which starts at conception and continues through production, testing, and operation (INCOSE 2004). The SE process is a means to achieving the customer's needs by determining the systems scope, objectives, feasibility, functions, requirements, solution alternatives, and concept of operations. Often, changes are made to these aspects of a project after the deadline for determining them. This is in partly due to a lack of understanding in regards to the overall user needs and the system functions by the project team and developers. A change to aspects of the associated systems engineering can result in schedule delays and cost overruns. Government Accountability Office (GAO) has reported that 55% of projects that have been re- lined are due to changes in the requirements, objectives or scope of the project (Government Accountability Office [GAO] 2008).

An example of a lack of thorough systems engineering can be seen in the Federal Bureau of Investigations (FBI) attempted acquisition of the Virtual Case File (VCF) that would replace the FBI's Automated Case Support (ACS) system which had become "cumbersome, inefficient, and limited in its capabilities" (Howerton 2009). One of the major contributing factors to the project's failure, as stated by Mr. Glenn A Fine, US Department of Justice Inspector General, was that the design requirements evolved slowly and were poorly defined (Office of Inspector General **[**OIG] 2005). Project requirements for the VCF system changed because the FBI's organizational priorities shifted to the prevention of terrorist attacks after the events of September 11, 2001 (OIG 2005). This shift resulted in a change in requirements, capabilities, and functionality. The FBI did not have a plan or process model to guide them through development, therefore when different aspects of the systems engineering work changed there was no way to integrate those changes into the project with any success. In the end, the VCF

system was cancelled in 2005 at a cost of $170M with no tangible product (Eggen and Witte 2006).

## 4. Requirements

In order for the customer to benefit from a software system a list of required functional and nonfunctional requirements need to be derived. Both the functional and nonfunctional requirements must be well founded and understandable for the delivered system to meet the needs of users (Debono 2012). Functional requirements describe what the system functions must actually do. The non-functional requirements describe how the system will perform. Functional requirements can also be viewed as inputs that result in outputs from the system. Nonfunctional requirements are performance attributes such as interoperability, reliability, sustainability, availability, and usability. From the functional and nonfunctional requirements, a specification is derived for a contract, which is then followed by a contractor to build the system. If these requirements are poorly written the resulting product will also be poor.

Some system projects write extensive requirements, while some projects have a very lean set of requirements, which results in over overbuild or underbuilding of the software. Overbuilt software contains many functions and capabilities needed to perform the original scope and stakeholder needs for the system. Underbuilt software lacks the functionality needed by users. Studies have shown that the largest area of software waste was due to over and under building the software (Standish Group 2009). While underbuilding software results in a system that delivers less than what is needed by users, overbuilding software results in money expenditure on unnecessary features. In fact, statistics have shown that as much as 50% of requirements requested features for software that was not utilized by users of the system (Standish Group 2009). To ensure the requirements are not too robust, or lean, skill must be taken to ensure each requirement maps to the needed functionality.

Another reason requirements present problems in system development is the requirements tend to change or grow while the project progresses. Changes in desired system functions and capabilities results in newly derived requirements. Requirement

changes, while often from project teams not fixing the requirements earlier in the process, are most often associated with the desire "to automate domains that are only partly understood" (Jones 1996, 92). While building software, these domains are uncovered and the requirements must change: in large-scale projects this can prove too big a difficult task to complete. The CG's previous attempt at a system produced a system that contained over 231 requirements of which many were not needed or too specific. Software development results in new domains being revealed, to dictate every requirement down to the lowest level of hierarchy can be impossible to do accurately. The CG's requirements for an MRD system should be fixed and detailed at the high level prior to entering development, but then free to be developed on lower, more specific levels as the project progresses.

### 5. Test and Evaluation

Software requires testing and evaluation to ensure the system delivers the needed capability, demonstrate system functionality, and is able evaluate the usability of the system delivered. Software testing needs to go beyond checking for errors and bugs. In an article concerning software testing, Jiantao Pan, claims that software testing is an "art" because not all aspects of software are fully understood (Pan 1999). Pan goes on to say that testing includes more than just debugging but also includes quality assurance, verification/validation, and reliability as well as testing being based on budget, time and quality (Pan 1999). Testing and evaluation can be a time consuming evolution and with an increase in project size comes an increase in the amount of testing and evaluation needed, which will increase the cost and the risk.

It is impossible to perform software testing on every input-output value. Software can have hundreds, to millions, of lines of code. Testing all possible combinations could take an exhaustive amount of time. In addition, for every addition input-output tested, more money and time in the schedule is required. There are also multiple types of testing that can, and usually needs to, be completed, including performance testing, white-box testing, black-box testing, reliability testing, and security testing (Pan 1999). A common guidelines used in determining the conclusion of testing is to stop when time or budget

are exceeded, when prescribed tests are complete, or when the reliability requirement is met (Pan 1999). Regardless of which tests must be performed and the method used to conclude testing, the testing needs to be manageable. The goal should be to keep the project as small as possible while still delivering the required capability. To do this a process model that has many small projects within a larger project could be beneficial.

An organization that choses to pursue a very large-scale IT system can learn valuable lessons from the consequences suffered from previous company's attempts. A large-scale example can be seen from the Northeast blackout in 2003. In August 2003, a large blackout affected areas of the Northeast and Midwest, and extended as far as Ontario with some areas not regaining power for up to two days *(Wikipedia* 2014b). After an investigation into the cause of the blackout, it was found that there was a bug in GE Energy's system software that caused a failure of the alarm system; thus leading to the events that allowed the blackout (Poulsen 2004). Company spokesman Ralph DiNicola was quoted as having said "it took them weeks of poring through millions of lines of code and data to find it" (Poulsen 2004). GE had created a large system with millions of lines of code and could therefore not test all combinations of code prior to the system being operational. Had the system been delivered in smaller increments, with testing performed on each increment, the error may have been found during the testing and evaluation stage of the project.

Testing and evaluation can take up large portions of a project budgets, especially if testing does not remedy the bugs and errors in a system and does not improve the quality or effectiveness. If testing methods in relation to software were improved, a resultant $22.5B could have been saved of the approximate $59.5B each year spent on software bugs (Thibodeau 2002). The challenge is to ensure accurate time is dedicated to the testing and evaluation of a system. Additionally, the system must be tested in ways beyond debugging and error checking, it must be tested for on the basis of quality, effectiveness, and reliability. Products delivered with extensive bugs or errors in the software will have a negative impact to customer satisfaction and return on investment for stakeholders.

The increase of software's complexity directly results in an increase in the number of lines of code. The bug to lines of code ratio is "about 15-50 errors per 1000 lines of delivered code" (McDonnel 2004). To keep the lines of code numbers low, it is important to ensure the system code is written to meet the functions required and nothing further. Additionally, by delivering a system in smaller increments, each increment can be tested quicker, more accurately, and provide greater chance of finding the maximum number of bugs/errors. While it was not the most significant challenge the CG had in its previous attempt at a usable MRD system, it was amongst the other challenges faced. The testing and evaluation plan, as prescribed in the MRD AIS Development and Support Plan, Version 2.3, called for testing to be performed but there was no written test plan or description for the types of tests (CG-1B1 2008). Without a test plan or the list of tests to be performed no measures of functionality or capability could be evaluated.

### 6. Risk Analysis

Risk analysis presents the final challenge of significance. Risks are those events that, if happen, will cause problems with a systems development or acquisition. All the challenges stated herein are areas where there is risk. Failure to perform risk analysis in will have consequences that can hamper or cancel a project. At the beginning of the project, most of the important decisions concerning a system are made. It is therefore important to begin risk analysis at conception because it is the "first, and perhaps best, opportunity to reduce acquisition risk" (GAO 2010). While risk analysis begins at project conception, it should not stop but continue through development, architecting, creation, and deployment.

Risk management for IT systems is crucial because of the complexity of technology. While some hardware aspects have slowed in evolution, software has not. A large contributor to project and system risk is the size of the system, the larger the system the more complex (Durkovic and Rakovic 2009, 15). A more complex system contains more lines of code, the internet, servers, networks, applications, databases and more through which risk is introduced (Chodhury and Arefeen 2011). Risks cannot be eliminated, only mitigated. To reduce the introduction of risk, the system may be built

incrementally, allowing risk to be managed in smaller batches. A process model that develops and architects the system in increments can better enable project teams to mitigate the foreseen risk. Unmitigated risks can cause one or many project issues such as late product delivery, cost overruns, lack of system capability or functionality, a failure to meet system requirements, and poor product performance.

Risk was high during the CG's last attempt to develop and acquire a MRD system. A failure to foresee that the system created only delivered the prescribed functions but no capability, due to lack of a user work flow, resulted in an unused system and therefore project funding waste. The CG project team was also not technically experienced in acquisition of IT system development, which left the project team unable to see that the system being developed would not meet their needs. If the CG was to choose to develop the MRD system by means of a process model that delivered the entire system at once, at the end of the project, it will be introducing risk from the onset. Process models that develop and architect systems in smaller less complex increments have a higher rate of success. Development in smaller increments allow risk analysis to focus on one or two components or capabilities at a time when performing testing and evaluation, determining budget and schedule, and when accomplishing the systems engineering aspects of development beginning at conception.

## C. SYSTEMS ENGINEERING AS A MEANS OF MITIGATING CHALLENGES

Since the 1990s there have been numerous reforms aimed at lessening the burden of acquisition challenges and preventing common mistakes. Reforms such the Federal Acquisition Streamlining Act of 1994, the Clinger-Cohen Act of 1996, and various other reforms legislated yearly through the national Defense Authorization Act have been used to mitigate risk and failure, yet IT projects routinely come in over budget and behind schedule. In 2012 it was reported that six DOD projects, which included the Air Force Defense Enterprise Accounting and Management System (DEAMS) and the Army's Logistics Modernization Program, were behind schedule and all together over budget by $8M (Reilly 2012). Acquisition reform has helped but not significantly improved project outcomes and mitigation of IT development challenges.

To mitigate the challenges associated with developing the MRD system, including the system complexity and risk, systems engineering analysis needs to be applied to all projects. The MRD systems engineering work will include a system scope, context, capabilities, and functions. This will be further discussed in Chapter Four of this report and show the importance of their inclusion in this project. While the systems scope, context, capability, and function identification is vital for system development, a means of governing and visualizing the system is equally or possibly more, important depending on one's perspective. For this report, the process model and system architecture are vital to development success and mitigating the challenges previously discussed.

### 1. Model-Based Systems Engineering

Model-based systems engineering (MBSE) is the use of models by systems engineers to limit the amount of written documentation and provide a visual representation of system engineering components (Giachetti 2010). With an appropriate process model, the system being developed will have a strong base, which will provide a "strategic advantage" during development over projects without a process model (Bredemeyer n.d.). The model will provide the development team with a means by which to simplify the system, manage the inherent software complexity, ensure team members understand the required system, and provide for traceability of capabilities and functions (Giachetti 2010). While a process model is needed, it is important to note that it must be an appropriate one, based on the type and key aspects of the envisioned system. When selecting a process model, the system should be analyzed to ensure alignment between the business, information, application and infrastructure areas of the enterprise and new system, and the software/hardware architecture being considered (Schafrik 2011).

The CG can benefit by using an appropriate process model for the development of a MRD system. The CG is no different than any other organization within the government and has been plagued by the challenges presented herein concerning IT system development and acquisition. Congress' Federal Information Technology Acquisition Reform Act (FITARA), passed in 2013 and updated in 2014, included

provisions for Departments within the government to use merit versus least cost when making choices during acquisitions (Congress 2013). Specifically, FITARA allows for selections to be made "based on performance and value" and be "free of preconceived preferences based on how technology is developed," and include the "consideration of proprietary, open source, and mixed source software technologies" (Congress n.d.). The CG's use of a process model is the best means by which to ensure it makes choices based on the provisions within this act. Overall, it could be used as a means of exploiting these provisions to ensure the CG truly receives the best system.

The CG cannot eliminate all potential risk while developing and acquiring a MRD system but it must mitigate them through the best means possible. Acquisitions have evolved throughout the last 24 years but the means of input into a system has not. The inputs leading to a system must be complete and accurate if acquisition is to be successful. A process model can mitigate IT development challenges but only if the appropriate process model for the project type, organization, and development team's level of experience is selected. There are many process models available for selection; four of them will be evaluated for this project.

### 2. System's Architecture

In addition to the process model, a MRD system's architecture has been developed to mitigate IT development challenges. The system architecture is a blueprint that guides the development team by visually representing the components necessary for the MRD system and how they are integrated to form a MRD system capable of delivering the needed functions. The system components were determined by selecting those that would be familiar and easy to use as well as easily integrated and replaced if needed. The system architecture incorporates systems and hardware already in use within the CG such as existing databases, the CG Network, and user interface hardware, such as laptops and desktop computers. The architecture developed for the MRD system is broken into smaller blueprints for each increment, which allows the final three-tier architecture to evolve while development moves forward through the process model.

THIS PAGE INTENTIONALLY LEFT BLANK

# III. SELECTION OF A SOFTWARE DEVELOPMENT MODEL

## A. INTRODUCTION

As demonstrated in the previous chapter, to accurately develop a system and ensure it meets user needs, a process model is required to guide system development. A process model ensures accurate execution of system development including the modeling both transition and target system architectures. The model contributes to the understanding and management of all stages of systems development and ensures it includes all required system functions, capabilities, performance goals, analysis of alternatives, cost, schedule, and risk analysis needed. There are a many different process models from which to choose when beginning a new IT System project. There is no one type of process model that can be used for every system due to the different needs and requirements for each system (Wang and Yang 2012). Therefore four different process models were considered for the MRD system to ensure a successful system development.

## B. WATERFALL MODEL

### 1. Overview

The waterfall model was developed in 1970, further revised in 1981, and consists of a series of phases, activities, or steps (Blanchard and Fabrycky 2010). Most often the model contains six common phases; requirements analysis, design, implementation, testing, installation, and maintenance (Hurricane Softwares n.d.). A graphical representation of the waterfall model is shown in Figure 1.

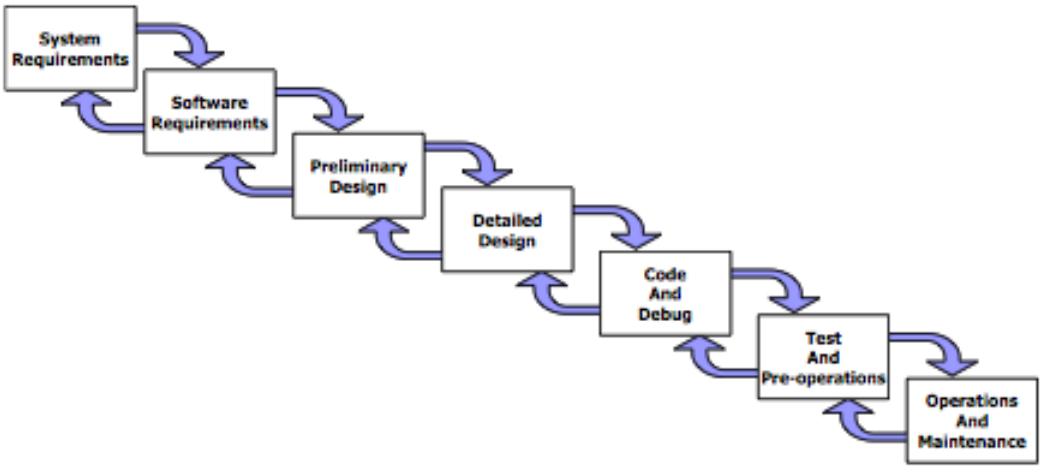


Figure 1. Waterfall System Development Model (from California Department of Transportation 2007)

While the model flows from one phase to the next such as a waterfall, it contains built in feedback loops that allow the project team to return to the previous phase of the project to make changes. The waterfall model is most often best used when the requirements are well known at the start of development, the system is simple and will not evolve, stakeholder involvement is not needed beyond the requirements phase, and control gates to future phases are not obvious (California Department of Transportation 2007). Requirements are composed in the beginning phases and once the system progresses beyond the stage following requirements, there is no method for feedback to the requirements phase. This makes the model even more difficult to use if the requirements are written before the customer or user knows exactly what is needed to solve their needs. What the customer wants usually becomes evident over the course of the project (Melonfire 2006). In an IT system this is even more so as an IT system with software is attempting to write code that automates a process or even a method humans use to accomplish an objective.

The waterfall model allows for testing but not until the end of the project. Late project testing implies that the entire system has been built, or software has been coded, before the system is testing against the requirements. This could cause major re-work and delays if the system does not meet the requirements forcing the project team not only back into the coding and building phase, but possibly the detailed and preliminary design

phases also. These phases and the order they are in are shown in Figure 1. This late stage is also when the pre-operational testing or display of capability is completed. If the system fails to deliver the capability required, major rework would be needed within the requirements and/or design causing the project team to retreat back into numerous earlier phases. Returning to early phases of the model can cause the schedule and costs to overrun up to 100% (Royce 1970).

### 2. Waterfall in Respect to the MRD

For the CG's MRD system, the waterfall model is not the model selected for use. This model does not allow for the flexibility needed by CG-1B4 in developing the required need of a system. One of the primary needs of a system model is that is must allow for evolution of requirements. While CG-1B4 has a list of requirements previously documents from project efforts, the requirements are not stable because of the potential for changing project leadership and in organization leadership. Additionally, the expertise of the team is not in software therefore the requirements needed may not be well understood. This will also be a reason for the need to include user feedback throughout the process, not just in the early requirements and then later testing phase.

Another disadvantage in selecting the waterfall model for CG-1B4's project is that no working product is delivered to the organization until the end of the project. This is disadvantageous because there will be no capability increase within CG-1B4 until the end of the project for the system which could take years for the entire system be developed and acquired. This will also require funding for the project to be available and committed in full upfront. If at any point there is a lack of funding, which delays the development of the system, then there will be an even greater risk of technology significantly outpacing the project. CG-1B4 requires a low risk model with user participation in the solution resulting in the capability being delivered sooner rather than later, the waterfall model will not aid in this.

## C. "VEE" MODEL

### 1. Overview

Another model commonly used is the "Vee" model, this model stresses the relationship of the early phases on the later phases (California Department of Transportation 2007). The "Vee" model has a left and right side that is shown in Figure 2.

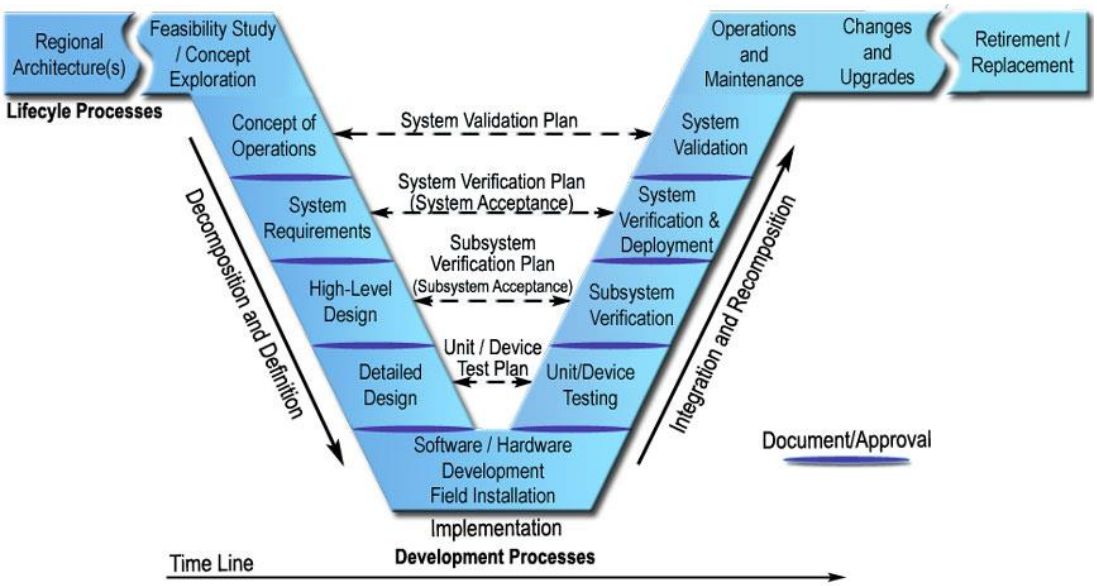


Figure 2. "Vee" System Development Model (from U.S. Department of Transportation 2007)

The left side of the "Vee" consists of the decomposition and definition development of the system to be built, with each level supporting the level below it (U.S. Department of Transportation 2007). As shown in Figure 2 by the dotted lines crossed the gap between the left and right side of the "Vee", the concept of operations will support system requirement, the development of the system requirements will then support the high level design and so on until reaching the bottom of the "Vee." The system development than continues up the right side of the model ending with system validation before moving on to post validation life cycle stages.

The "Vee" model advantages include decision gates between each level of the model, it visual representation demonstrates how each level on the left side of the model relates to the right side, and allows for stakeholder involvement on both sides of the model California Department of Transportation 2007). The "Vee" model was not chosen for this project largely due to the cost and resources needed to implement and execute this model, prototype software is not made, and it is not flexible in the event of change in the middle of the model (Admin 2009).

### 2. "Vee" in Respect to the MRD

This model fits the development of the MRD system for CG-1B4 better than the waterfall model but still falls short. The model may be easy to use and understand, dictates the deliverables of each phase, and develops test plans early in the project, it still has shortfalls that could cause higher risk than necessary for CG-1B4 (Munassar and Govardhan 2010, 96). Like the waterfall model, the "Vee" model requires a full understanding of the requirements from the beginning of the project and will not support any changes to the requirements during the model progression. With the possibility of project team change out and organizational leadership changes during the projects progression, CG-1B4's requirements have the potential for changing. In addition, because the system verification plan is derived when the requirements are written, shown in Figure 2 at the beginning of project progression, a change in requirements will negate the system verification plan causing a rewrite.

Two additional disadvantages of this model for CG-1B4 include the lack of prototypes developed during the model progression (Munassar and Govardhan 2010, 98) and the need for personnel with high technical expertise and resources (Federal Reserve 2007). CG-1B4 will need prototypes in order to ensure the product delivers the capability needed early in the project in order to reduce risk. Additionally, CG-1B4 does not have the software/IT technical expertise needed for a model that requires a high level of it. These points create the potential of a high amount of risk and uncertainty within this model for CG-1B4 and therefore this model was also not chosen.

## D. SPIRAL MODEL

### 1. Overview

The spiral model is "a cyclic approach for incrementally growing a system's degree of definition and implementation while decreasing its degree of risk." Each component, group of components, of the system, be it a database, application, report generation, or software, is built during a cycle or 360 degree rotation around the model. Each iteration about the circle results in a usable capability for the stakeholder. The shown in Figure 3 represents a common version of the basic concepts and cycles used in a spiral model.

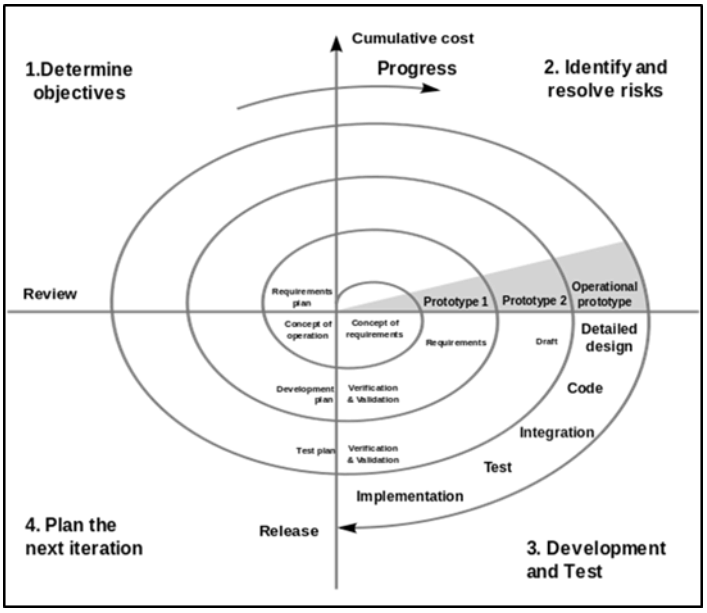


Figure 3. Spiral Model (from Boehm 1988)

The spiral model focuses on four key phases within each iteration including planning, risk analysis, engineering, and evaluation (Munassar and Govardhan 2010, 99). The advantages of the spiral can be seen from these four phases. Within the planning phase requirements are developed, simulation and models can be developed, and development or integration plans can be developed, Figure 3. The next phase is risk where the risk is not only analyzed but can be analyzed based on the current state of the system, or iteration, which allows for a more accurate up to date analysis of the project.

Additionally, the risk phase allows for identification of alternative solutions moving forward in each iteration of the model (Munassar and Govardhan 2010, 98). In the engineering and evaluation phases, prototypes are developed, which allows the users to see the capability being delivered incrementally, and then test that prototype against the requirements for that iteration (Munassar and Govardhan 2010, 99).

### 2. Spiral Model in Relation to MRD

At first glance this appears to be the ideal model for the CG's development of a MRD system. The Spiral model places an emphasis on risk analysis throughout system development mitigating many of the challenges and mistakes that can be faced in software development. The system also enables prototypes and testing throughout the project and the development of requirements for each future iteration from the capability delivered in the previous.

While a significant amount of risk analysis is appealing to many organizations, it requires personnel with a high level of expertise in risk filed to be assigned to the project for its entire duration. Additionally, it does not always work best for projects that are smaller to incorporate a significant portion of time to risk analysis (Munassar and Govardhan 2010, 100). CG-1B4's project is smaller in that it is not a billion dollar system, an enterprise wide system, or a system with a development schedule need of more than 2 years. In addition to an overabundance of risk analysis for the MRD project, there is a lack of feedback paths to aide in ensuring a usable system is delivered.

## E. ITERATIVE INCREMENTAL DEVELOPMENT MODEL

### 1. Overview

The final process model explored is the Iterative Incremental Development (IID) model. This model is based on an overall lifecycle that is made up of many iterations or "mini-projects" with each resulting in a "subset of the final system" (Larman 2004). An example of a simplified view of the concepts contained within an IID process model is shown in Figure 4.

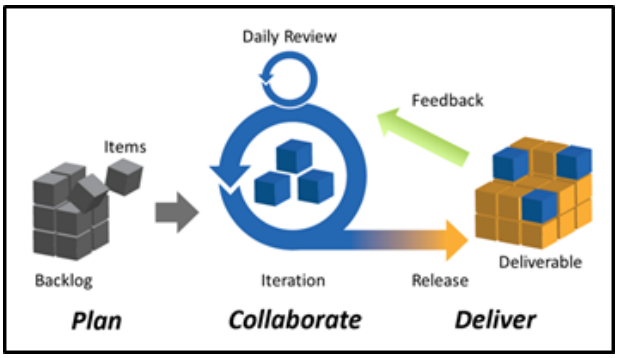


Figure 4. Simplified view of IID (from *Wikipedia* 2014a).

Each iteration of the system within this model can be both risk driven an organization driven based on internal priorities and can contain engineering aspects such as requirements, design, programming and so on (Larman 2004). The model is based on the belief, of a given system development, that not all aspects of the system are truly known at the project onset and feedback is given early in order to ensure the meeting of user requirements (MacCormack 2001, 78). An example of the IID process model from IBM, displaying multiple iterations is shown in Figure 5.

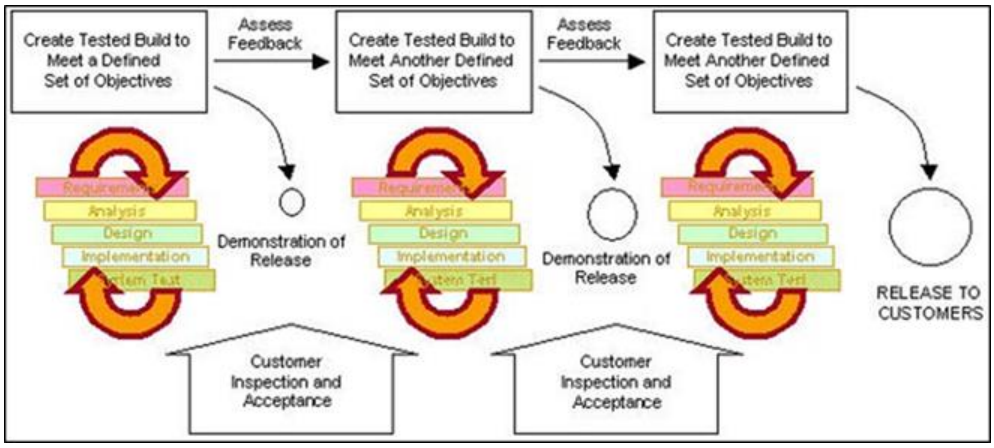


Figure 5. Overall IDD Concept (from Spence and Bittner 2005).

Each iteration shown in Figure 5 is time based, which is called "time-boxing," and used to fix the start and end date. The time boxing is usually based on weeks to months with a goal of not exceeding six months. By limiting the time to six months for an iteration it lowers the risk (Larman 2004). These iterations are delivered incrementally

to the organization, with each one expanding on the capability of the system delivered in previous increments, therefore extending system capability at each stage.

When comparing the waterfall model against the IID process model, the IID process model is the better option. Research conducted on the waterfall and IID process models showed that IID had lower risk and better project outcomes including those associated with defect rates (Larman 2004). A comparison graphic of the Waterfall and IID process model can be seen in Figure 6. The figure depicts the waterfalls classic set up of requiring requirements and design being upfront in the process, which limits change throughout the systems development versus the cyclic approach of the iterative process.

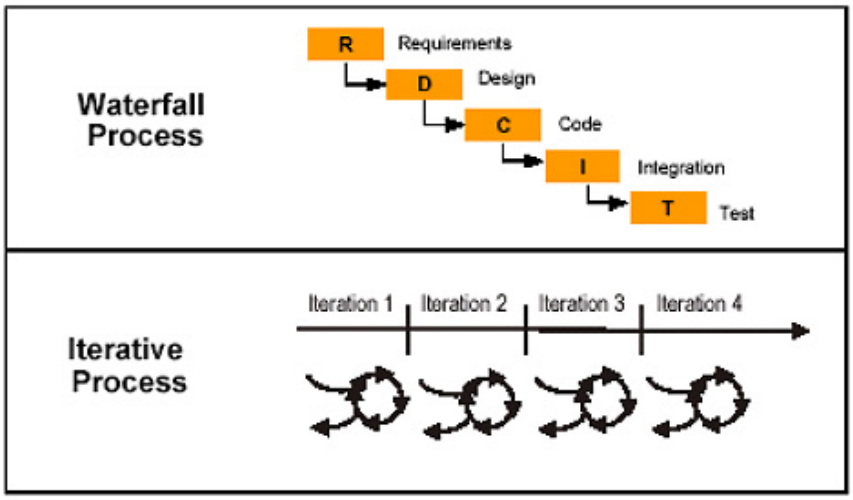


Figure 6. Waterfall Model versus Incremental Iterative Development Model (from Bittner 2006).

While the waterfall model continues to be promoted, evidence suggests that it is not ideal for software development. A study conducted on the Internet-software industry by MacCormack discovered that "getting a low-functionality version of the product into customers' hands at the earliest opportunity improves quality dramatically" which is not achievable when using the Waterfall and most other models (MacCormack 2001, 79). The IID model allows for an initial build of capability and follows that with additional capability delivery in each iteration of the project. The model takes the previous version of work, analyzes it, and then reworks it in order to add capability (Fairley and Willshire

2005). Advantages to this include repeated analysis of the system and integrating changes that result from that into the next iteration (Fairley and Willshire 2005).

While the spiral model allows for evolution of requirements, the IID model does not, and is based on "stable requirements" for each iteration (Fairley and Willshire 2005). When beginning a new iteration, there is room for validation of the requirements before freezing them again for iterations completion, shown in Figure 7.

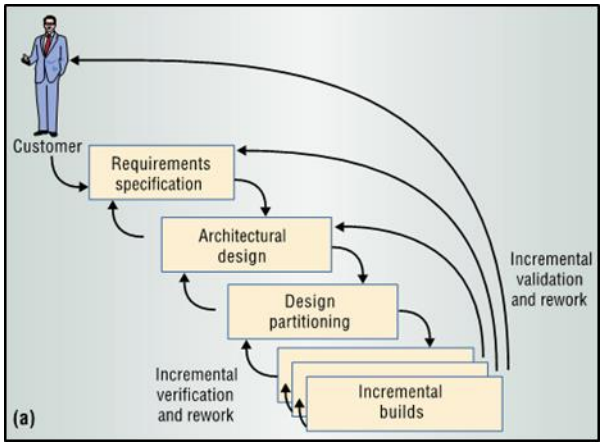


Figure 7. Build sequences as presented by Fairley and Willshire (from Fairley and Willshire 2005)

During each incremental build rework of the previous build can be completed based on feedback gained from the users allowing for a graceful modification of the product (Fairley and Willshire 2005). This eliminates the common schedule and budget problems caused by other models that do not allow for modifications throughout the product evolution. It is important to note that there can be too much and too little rework when using this model. During the first phases rework can be greater than later phases, as "output from each phase stabilizes" rework lessens (Cauwenberghe 2002, 15).

### 2. IID in Relation to MRD

Due to the advantages of the IID model, the nature of the MRD system, and the personnel realities of CG-1B and CG, the IID model has been selected as the model of choice for developing the MRD system. The biggest advantage for CG-1B4 will be seen

from the process model's overall structure, which will guide the development team into each future iteration, yet provide them flexibility when building the capabilities within each iteration. The model will allow CG-1B4 to select the overarching requirements from previous project analysis and apply them to this project, yet allow them to change and be broken into smaller requirements as needed throughout development. This will reduce the risk of heavy rework and schedule slippage by allowing time for rework within the schedule. The main means for translating changes to the requirements is through feedback opportunities that are built in and integral to the IID process model.

The IID process model has a dedicated and integrated feedback path, which allows CG-1B4 to take advantage of user participation. The IID process model's built in feedback ensures feedback both on the previous iterations capability delivered and feedback aimed at ensuring the next increments capability is validated. Previous attempts at an MRD system did not allow feedback to be a valued aspect of the project, often being lost or not included by developers. The other benefit is that the upfront input from organization can be changed in later increments. If personnel within the project team, or leadership, changes direction with their vision or needs for a MRD system, the next iteration can take that feedback and apply it to make the changes needed.

Another benefit for the CG is that this process model will also allow for the system to mature over time reducing faults/bugs resulting in a more reliable system (Sotirovski 2001, 72). Systems that are large tend to have more code and subsequently more bugs. Smaller increments of a system will allow for better remedy and identification of system bugs before the system gets too large. When the system is delivered in these smaller increments, CG-1B4 analysts will have increased workplace productivity due to having use of partial system functionality, from the completion of the first iteration and onward. This usage then goes back to again support system maturity benefits and feedback benefits. The users can run the system through real day to day usage allowing for errors to present themselves and future capability identification to take place (Sotirovski 2001, 68). Additionally, while the system is maturing, each iteration will leverage the software and hardware technology that is evolving in the public domain, which will aid in reducing the likelihood of the software becoming outdated. Finally, the

IID process model is scalable and can therefore fit a project that is not as large as an enterprise wide system but not so small not to affect multiple components of an organization. The next step is to perform a needs and requirements analysis for the MRD system in order to ensure an accurate IID process model development and systems architecture.

# IV. UP-FRONT SYSTEMS ENGINEERING ANALYSIS

## A. INTRODUCTION

The up-front systems engineering analysis will ensure that the MRD system developed, through the MRD IID process model, and the three-tier system architecture reflect the true needs, requirements, functionality, and capability needed by the CG. This chapter will analyze the overall scope of the project, assumptions and constraints, functions, capabilities, requirements, and system components. The system scope will allow us to define what will be included as part of the system and will require development and what is not within the scope of development for the MRD system. The system context will demonstrate how the MRD system will exchange energy, matter, material wealth, and information (EMMI). An understanding of the user's needs leads to derivation of the functions and capabilities required as well as what components may be capable of delivering them. The up-front systems engineering analysis will be used by the CG as guidance for development of the system while performing the activities within the IID process model and as justification of the developed three-tier systems architecture.

## B. SYSTEM SCOPE

The scope of the MRD system includes the software and hardware required to deliver the functions and capability stated herein. In addition, the scope includes all firewalls and security measures necessary for the MRD system to operate within the CG network/enterprise. Outside of the MRD system scope are all business operations, conducted by CG-1B4 not in direct relation to completing MRAs and MRDs.

## C. CONTEXT

The MRD context has five major organizational elements, each one providing input/output to the system, in the form of information. The five organizational elements exchanging information with the MRD system are CG-1B4 Analysts, MRD Sponsors, external CG organizational elements, CG Databases, and external system CG data. The

organizational elements exchange of information in relation to the MRD system is shown in Figure 8.

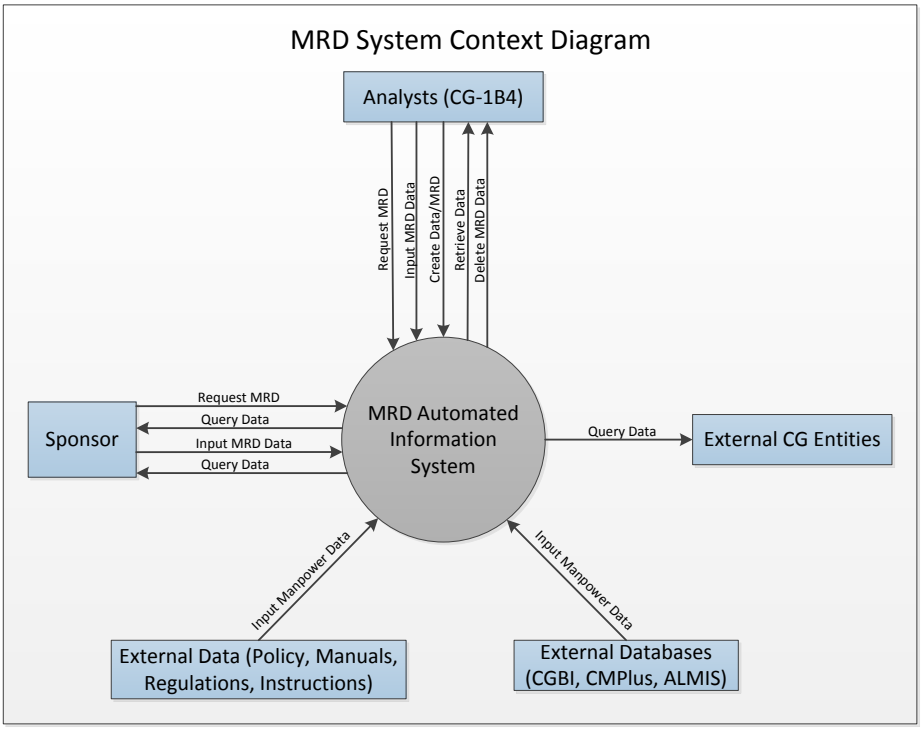


Figure 8. MRD System Context Diagram

Each one of these organizational elements is a classification of user types who will access and use the system in the same manner. The organizational elements are defined as follows:

- CG-1B4: Analysts working and assigned to CG-1B4 to include managers, supervisor, and administrators. In addition this includes analysts at NAVMAC and contractor personnel hired by CG-1B.

- MRD Sponsors: The MRD sponsors are the individual unit that is the sponsor or requester of the MRD.

- Eternal CG Organizational Elements: This includes all CG elements outside of CG-1B4 including project managers, program managers, and other individuals requiring workforce data for the completion of work outside of CG-1B4 and MRDs.

- CG Databases: System databases external to the MRD system that provides supplemental data for the MRD system database.

- CG External Data: policies, manuals, regulations, and instructions that have impact and govern manpower within the CG.

## D. CAPABILITIES

The MRD system's IID process model and system architecture were derived from the CG's needed capabilities through analysis and review of the CG Staffing Logic and Manpower Requirements Manual, COMDTINST M5310.5, Chapter II, Section A, B, and C. The overarching system capability is the ability to accomplish the MRA and subsequent MRD from Start to Finish, with limited input from analysts.

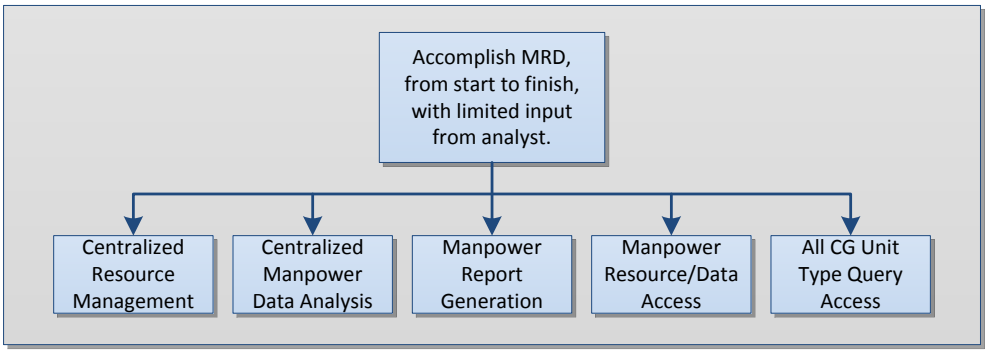


Figure 9. MRD System Capabilities

From the overarching capability of accomplishing MRDs, five sub-capabilities were derived to breakdown the overarching capability into smaller manageable capabilities:

- Centralized resource management
- Centralized manpower data analysis
- Manpower report and model generation
- Manpower resource/data access to all CG organizational elements
- All CG unit type query access real-time manpower data access

The sub-capabilities then allow for ease of mapping between capabilities and functions and capabilities and components. Each capability is described in further detail below:

- Centralized Data Management: The system shall be capable of collecting, storing, sorting, and filing data fields and information related to manpower.

- Centralized Manpower Data Analysis: The system shall be capable of analyzing the data collected, stored, sorted, and filed in order to translate it into usable fields, templates, graphs, models and other output forms to support decision making.

- Manpower Report Generation: The system shall be capable of producing reports, both official CG forms and general data reports.

- Manpower Resource/Data Access to all CG organizational elements: The system shall be capable of providing information to CG unit personnel, outside of the Human Resource Directorate, for review and some input.

- All CG Unit Type Query Access: The system shall be capable of a "lite" version of the application in order to provide remote, afloat, and other field units with limited intranet access the means to access and input manpower data.

## E. SYSTEM FUNCTIONS

### 1. Overview

The system capabilities will be realized through system functions. System function's, as defined by Blanchard and Fabrycky, are the actions needed for the system to achieve an operation through the use of an input such as people, data, or equipment (Blanchard and Fabrycky 2010). The functions for the MRD system were derived using the Coast Guard's Staffing Logic and Manpower Requirements Manual, Volume II, which is currently in draft form (U.S Coast Guard n.d.). The MRD system has three high-level functions, Figure 15. In addition, each function is broken into further sub functions, Figures 16-18.

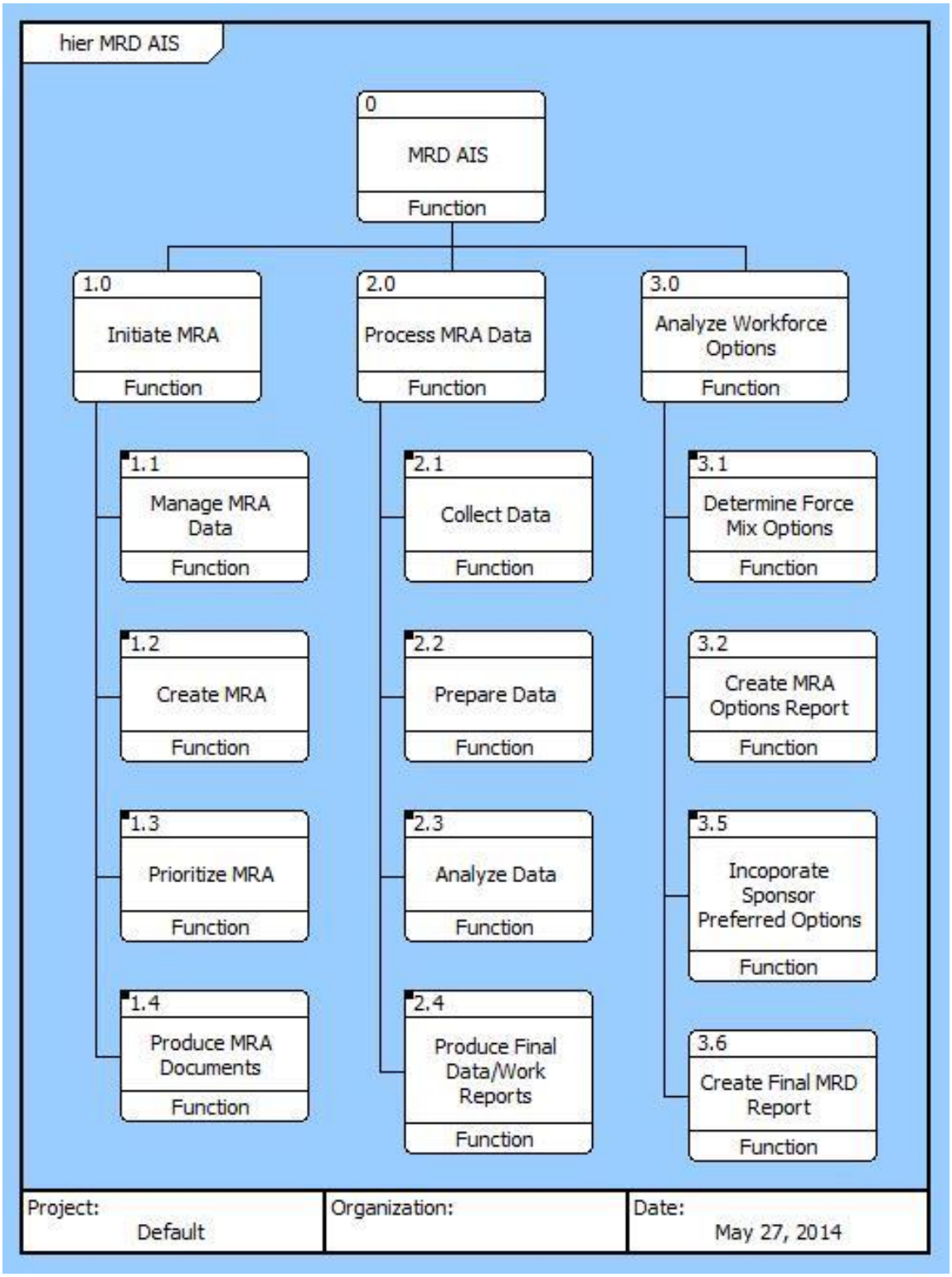


Figure 10. MRD System High Level Functional Hierarchy

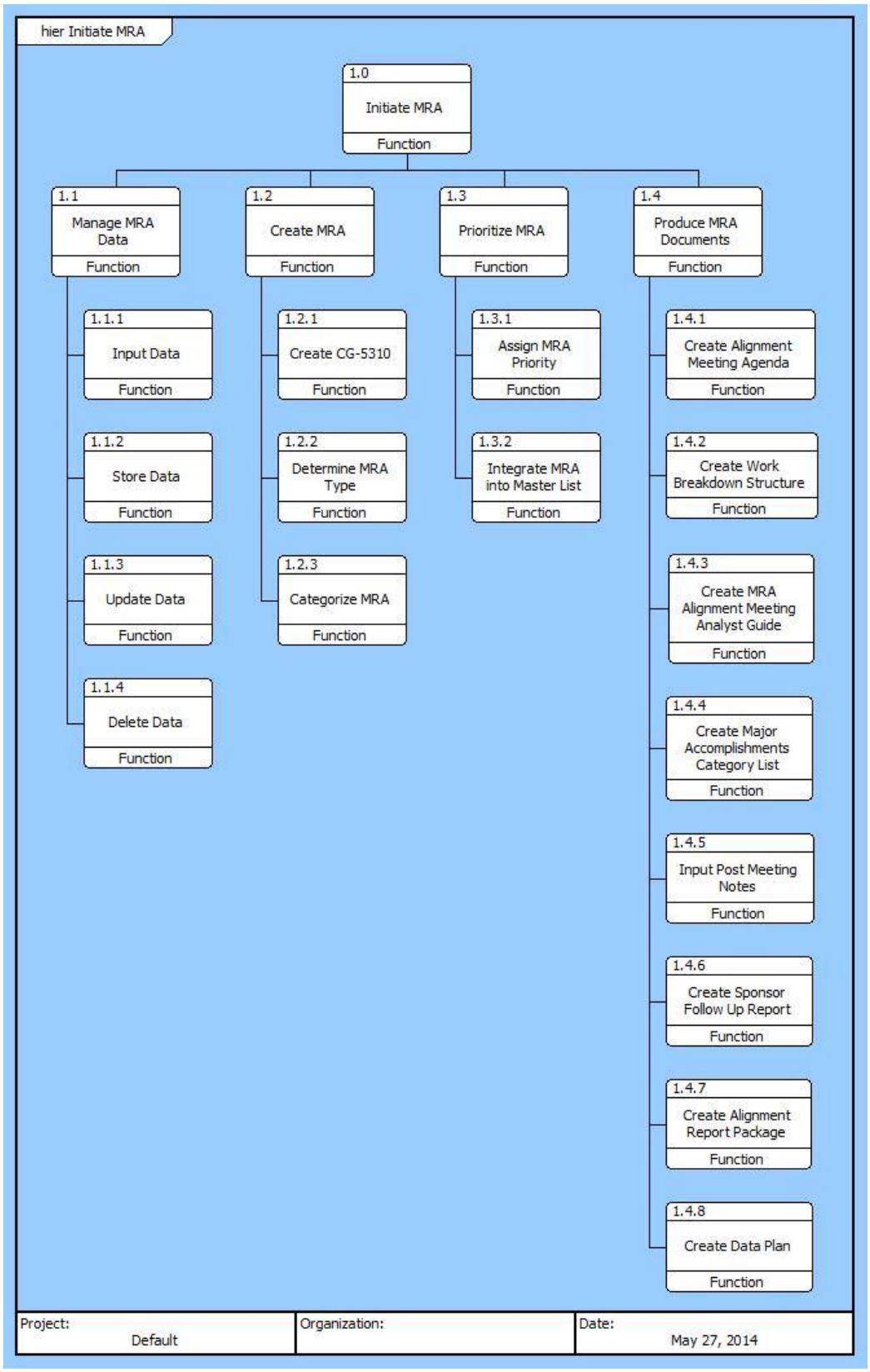


Figure 11. MRD System Function 1.0 Hierarchy

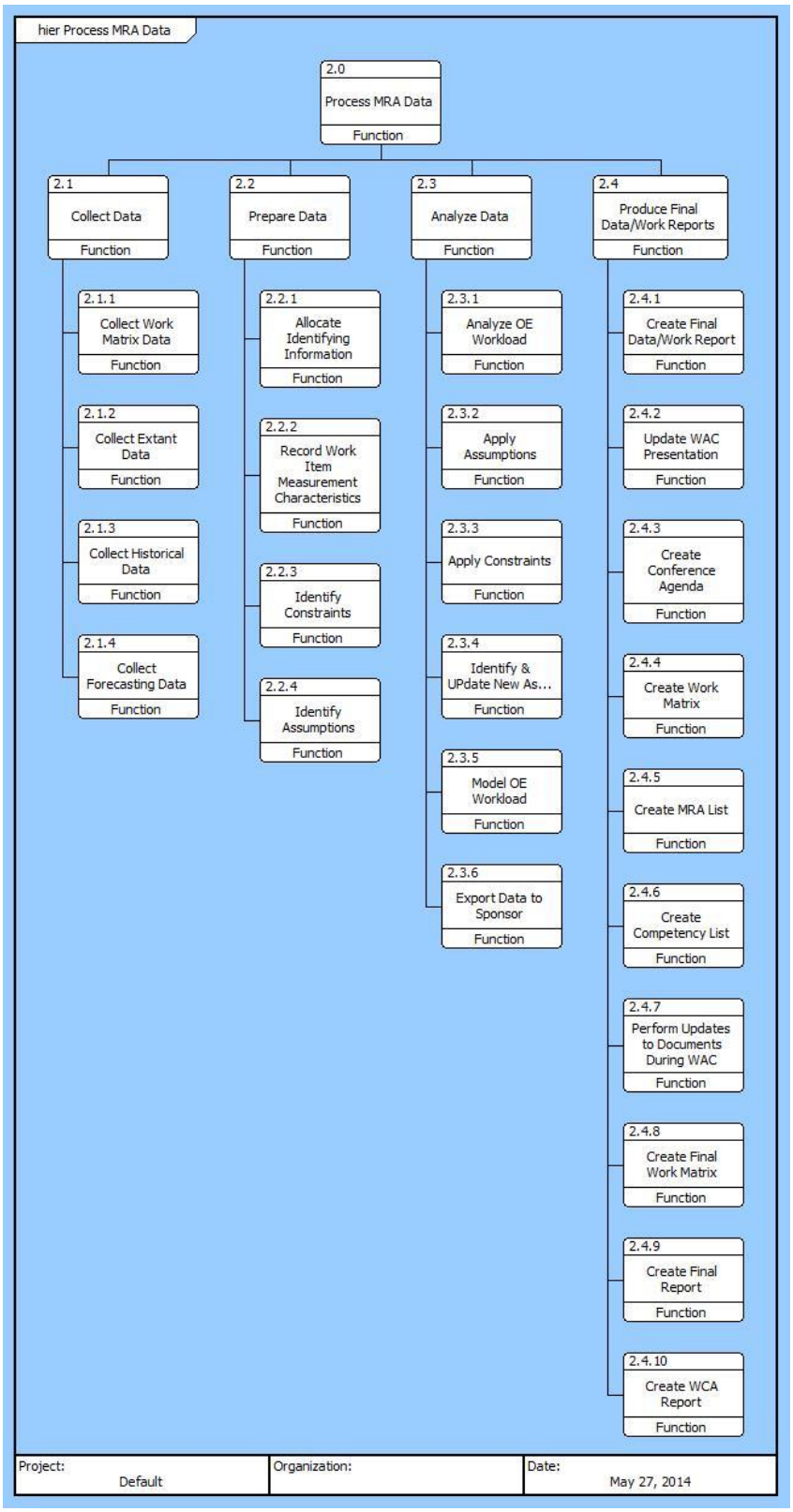


Figure 12. MRD System Function 2.0 Hierarchy

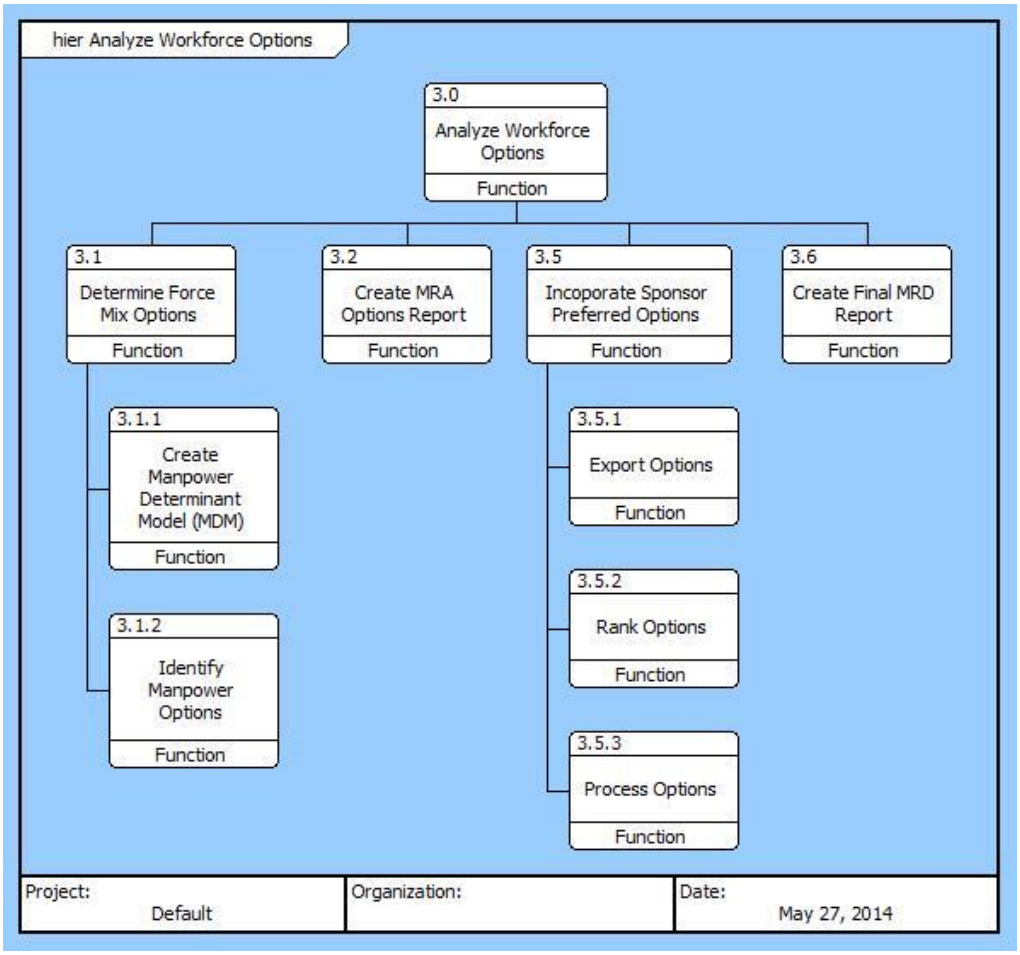


Figure 13. MRD System Function 3.0 Hierarchy

## 2. MRD System Functions

The first major function of the MRD system is to initiate the MRA and encompasses the analysis needed to move forward toward a complete MRD. The first function contains four sub-functions with associated further sub-functions, shown in Table 1.

| Function | | First Level Sub-Function | | Second Level Sub-Function | |
|---|---|---|---|---|---|
| 1.0 | Initiate MRA | 1.1 | Manage MRA Data | 1.1.1 | Input Data |
| | | | | 1.1.2 | Store Data |
| | | | | 1.1.3 | Update Data |
| | | | | 1.1.4 | Delete Data |
| | | 1.2 | Create MRA | 1.2.1 | Create CG-5310 |
| | | | | 1.2.2 | Determine MRA Type |
| | | | | 1.2.3 | Categorize MRA |
| | | 1.3 | Prioritize MRA | 1.3.1 | Assign MRA Priority |
| | | | | 1.3.2 | Integrate MRA into Master List |
| | | 1.4 | Produce MRA Documents | 1.4.1 | Create Alignment Meeting Agenda |
| | | | | 1.4.2 | Create Work Breakdown Structure |
| | | | | 1.4.3 | Create MRA Alignment Meeting Analyst Guide |
| | | | | 1.4.4 | Create Major Accomplishments Category List |
| | | | | 1.4.5 | Input Post Meeting Notes |
| | | | | 1.4.6 | Create Sponsor Follow Up Report |
| | | | | 1.4.7 | Create Alignment Report Package |
| | | | | 1.4.8 | Create Data Plan |

Table 1. MRD System Function 1.0 Decomposition

The first sub function of function 1.0 is manage MRA data (1.1) and enables the system to input, store, update, and delete data. The second sub function (1.2) is to create the MRA document, which begins the analysis and allows for selection of a MRA type and category. The next sub function (1.3) is to prioritize the MRA, which places the MRA onto the master list of MRA's to be completed and allows for the analysts to manage all of the MRA's in the system. The fourth sub function (1.4) provides for the production of alignment meeting documents. This sub function aids the analysts in production and completion of the numerous reports needed for completion of the MRA.

The second function (2.0) is to process MRA data, which includes four sub functions (2.1 through 2.4), and aides in the processing of data including collect data, prepare data, analyze data, and produce final data/work reports. Function 2.0 is then

further broken into additional sub functions that allow for analysts to complete the work related to the MRA that is needed to complete the MRD, Table 2.

| Function | First Level Sub-Function | Second Level Sub-Function |
|---|---|---|
| 2.0 Process MRA Data | 2.1 Collect Data | 2.1.1 Collect Work Matrix Data |
| | | 2.1.2 Collect Extant Data |
| | | 2.1.3 Collect Historical Data |
| | | 2.1.4 Collect Forecasting Data |
| | 2.2 Prepare Data | 2.2.1 Allocate Identifying Information |
| | | 2.2.2 Record Work Item Measurements Characteristics |
| | | 2.2.3 Identify Constraints |
| | | 2.2.4 Identify Assumptions |
| | 2.3 Analyze Data | 2.3.1 Analyze OE Workload |
| | | 2.3.2 Apply Assumptions |
| | | 2.3.3 Apply Constraints |
| | | 2.3.4 Identify & Update New Assumptions/Constraints |
| | | 2.3.5 Model OE Workload |
| | | 2.3.6 Export Data to Sponsor |
| | 2.4 Produce Final Data/Work Reports | 2.4.1 Create Final Data/Work Report |
| | | 2.4.2 Update WAC Presentation |
| | | 2.4.3 Create conference Agenda |
| | | 2.4.4 Create Work Matrix |
| | | 2.4.5 Create MA List |
| | | 2.4.6 Create Competency List |
| | | 2.4.7 Perform Updates to Documents During WAC |
| | | 2.4.8 Create Final Work Matrix |
| | | 2.4.9 Create Final Report |
| | | 2.4.10 Create WCA Report |

Table 2. MRD System Function 2.0 Decomposition

Function 3.0, analyze workforce options, is the last of the higher-level functions required for the system and enables analysts to make determination about manpower based on analysis completed in function 2.0. The function has six supporting sub functions (3.1 through 3.6) including determine force mix options, create MRA options report, analyze viability of options, produce workload data diagrams, incorporate sponsor preferred options, and create final MRD report, shown in Table 3.

| Function | First Level Sub-Function | Second Level Sub-Function |
|---|---|---|
| 3.0 Analyze Workforce Options | 3.1 Determine Force Mix Options | 3.1.1 Create Manpower Determinant Model (MDM) |
| | | 3.1.2 Identify Manpower Options |
| | 3.2 Create MRA Options Report | |
| | 3.3 Analyze Viability of Options | |
| | 3.4 Produce Workload Data Diagrams | 3.4.1 Create Workload by Work Type Diagram |
| | | 3.4.2 Create Workload by Work Function Diagram |
| | | 3.4.3 Create Worklaod by OE Sub-structure Diagram |
| | | 3.4.4 Create Workload by Manpower Type Diagram |
| | | 3.4.5 Create Workload by MA Diagram |
| | | 3.4.6 Create Workload by Competency Diagram |
| | 3.5 Incorporate Sponsor Preferred Options | 3.5.1 Export Options |
| | | 3.5.2 Rank Options |
| | | 3.5.3 Process Options |
| | 3.6 Create Final MRD Report | |

Table 3. MRD System Function 3.0 Decomposition

### 3. High Level Non-Functional Requirements

The MRD system, in addition to functional requirements, will have non-functional requirements. The non-functional requirements to be considered by the development team include:

- Usability: The MRD system must be usable in order to delivery capability and functionality. The system should be simple to use with a logical workflow. The usability of the system is important to ensure that analysts can use the system easily with minimum training. Without this analysts may not be capable of using the system.

- Interoperability: The MRD system should be interoperable with the systems identified in the system architecture by means of interfaces. The system should also me interoperable with existing information system hardware. The interoperability of the system allows for integration of the system into the CG network. Interoperability within the system will allow for use of different hardware and connections.

- Security: The MRD system will require security measures as dictated by current CG information system security protocols. This is vital to support the non-functional requirement of interoperability and to function within the CG network.

- Supportability/Maintainability: The MRD system should be able of being supported and maintained by current IT system personnel. The current CG IT personnel should be capable of maintaining the system with basic IT repair knowledge. In addition, it is important the system be composed of off-the-shelf hardware to make obtaining and replacing components easy to achieve.

There are multiple ways to capture and ensure the system has the non-functional requirements. It is recommended that the CG utilize operational scenarios/use case, user test and evaluation during the build phase of each iteration, and feedback integration during each iteration.

## F. FUNCTION TO CAPABILITY MAPPING

Each function of the system is mapped back to the capability it provides. This mapping provides assurance that each function serves to deliver capability and that resources are not wasted in developing system functionality that does not provide this

capability. The high level function to capability mapping for the MRD system is seen in Table 4.

| Function | Capabilities | | | | |
|---|---|---|---|---|---|
| | Centralized Resource Management | Centralized Manpower Data Analysis | Manpower Report Generation | Manpower Resources/Data Access | All CG Unit Type Data Access |
| Initiate MRA | x | x | x | x | x |
| Manage MRA Data | x | | | x | x |
| Create MRA | x | x | x | | |
| Prioritize MRA | x | | | | |
| Produce MRA Documents | x | | x | | |
| Process MRA Data | x | x | x | x | x |
| Collect Data | x | x | | x | x |
| Prepare Data | x | x | | x | |
| Analyze Data | | x | | x | |
| Produce Final Data/Work Reports | | x | x | | |
| Analyze Workforce Options | | x | x | x | |
| Determine Force Mix Options | | x | | | |
| Create MRA Options Report | | | x | | |
| Analyze Viability of Options | | x | | | |
| Produce Workload Data Diagrams | | | x | | |
| Incorporate Sponsor Preferred Options | | | | x | |
| Create Final MRD Report | | x | x | | |

Table 4. Function to Capability Mapping

THIS PAGE INTENTIONALLY LEFT BLANK

# V. MRD SYSTEM ITERATIVE INCREMENTAL DEVELOPMENT MODEL

## A. OVERVIEW

The MRD system IID process model breaks the system development into seven iterations. Each iteration contains five activities: analysis, build, implement, deploy, and support. These activities create a logical and sequential manner of progressing through each iterations system development. Each iteration is between four and six months in duration, with an overall project duration of 18 months. Of the activities, the analysis activity will usually require more time then subsequent activities. The iterations are sequential, but are overlapping. When the development team completes the analysis activity and enters into the build activity, the next iterations analysis activity begins. Upon completion of each iteration system capability will be delivered, it will evolve as the MRD system evolves with full system capability delivered with iteration seven. A graphical representation of the MRD system IID model's iteration activities are shown in Figure 14. The IID process model in its entirety, including a timeline, is shown in Figure 15.

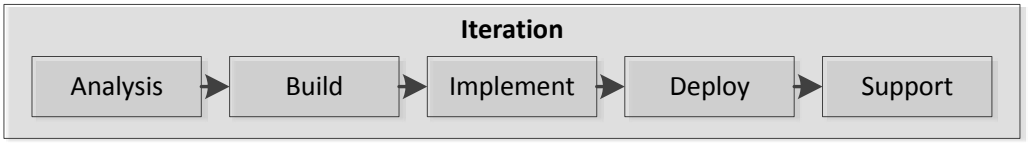


Figure 14. IID Model Iteration Activities

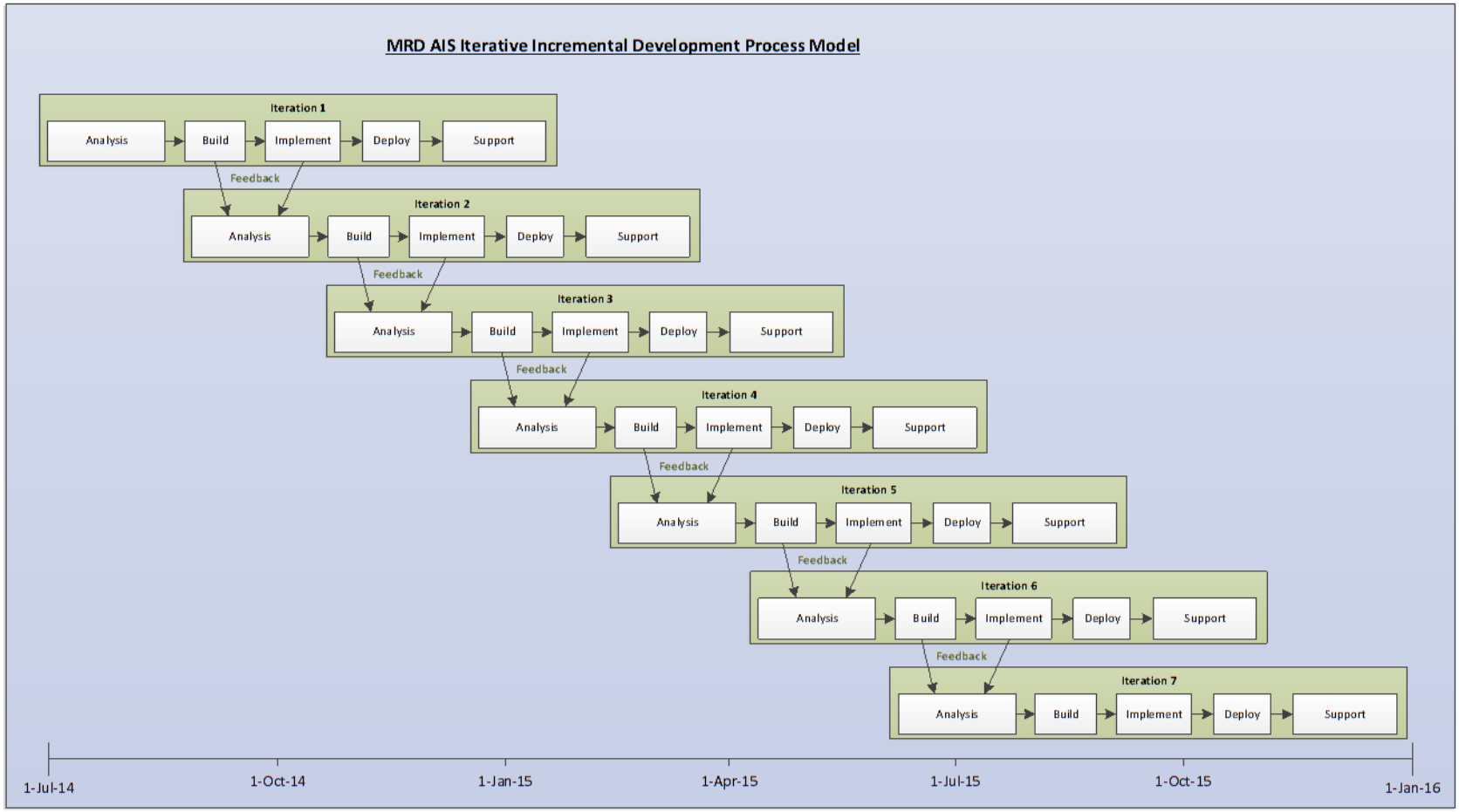


Figure 15. Coast Guard MRD System IID Process Model

While the process model breaks the system down into less complex iterations for development, a project manager should still lead the team. In addition, the development team should consist of representatives from CG-1B4, CG-6, and acquisition personnel. Having a robust and diverse development team, including a systems engineer, will enable the most effective use of the process model and ensure all activities within the iteration are carried out with maximum input and output.

During the analysis activity the development team will use the up-front engineering analysis completed in Chapter Four of the report to analyze the objectives, functional requirements, and capabilities to be delivered at the completion of the iteration. During analysis, the development team determines the specific goals, objectives, risks, budget, schedule, tasks, use cases, and resources needed for the given iteration. From this analysis, in particularly the functional requirements, a contract specification will be derived that provides the documentation from which a contracted company can build the system. The analysis activity is also the time during which the feedback form previous iterations, that have entered the build and implement activities, will be received for incorporation. It is important to note that once an iteration leaves the analysis activity, and enters the build activity, the project manager will not make any changes in the requirements dictated in the specification, but allow the build and future activities to continue on to completion (Larman 2004). If changes are realized after the analysis activity, those changes can be incorporated in further iterations. The product of any given iteration is not so great as to hinder the project if allowed to completely develop in spite of changes.

The second activity is the build activity, which commences upon completion of the analysis activity and is based on the specification developed during the analysis activity and the systems architecture for the given iteration. During this activity the developer, or contractor, is responsible for making the architecture a reality, building software, integrating hardware, and conducting all product testing. Once this is complete and the system is usable, the implementation activity begins. The implementation activity is the first time the product is “fielded” for use by the users. The developer will demonstrate the use of the system and the ways in which the system capabilities are

realized. The developer will also perform testing using use case scenarios with users, which will facilitate feedback for inclusion in later iterations of the system. The implementation activity also includes training for users specified during the analysis activity.

Once testing, feedback, and training are completed the fourth activity called deploy commences. This activity covers the time period where the system is handed to the users to use on a day-to-day basis. Once the system is deployed and being used, with little error or feedback, the iteration progresses into the support activity. The support activity lasts the length of time it takes to the next iteration. Once the deployment of the next iteration occurs the system is now "upgraded" or "updated" and has new support needs that begin anew when entering the subsequent activities support activity. The details of these activities, specific to each iteration are explained later in this report.

## B. FEEDBACK

As development progresses through each iteration, there are two initial paths of feedback incorporation. The first initial path of feedback is from the build activity, which feeds into the beginning of the next iterations analysis activity. The second path of feedback incorporation is from the implementation activity into the later evolution of the analysis activity. Figure 10 shows these initial feedback paths.

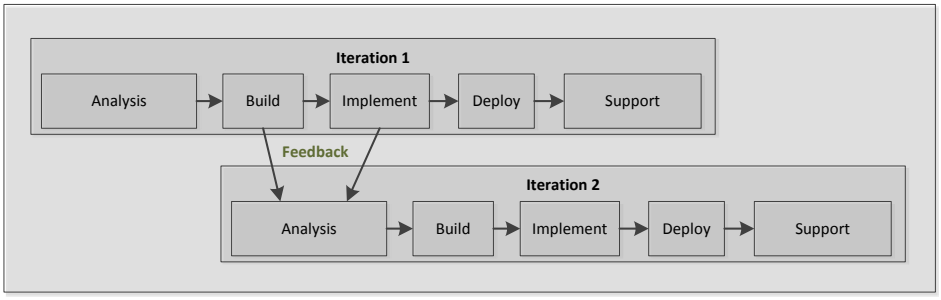


Figure 16. Iteration Feedback Lines

While not depicted, in Figure 10, to keep the model graphic uncluttered, there are additional feedback paths. Feedback paths are also present from the deploy and support activity. These paths do not feed into the next iteration but later iterations.

## C. SCHEDULE

The IID process model is a schedule driven model versus the traditional requirements driven models. Figure 8 incorporates a schedule ruler showing the project commencing on 01 July 2014 and concluding on 01 January 2016. This schedule can be edited to reflect any start date but the development of the system should last no longer than approximately 18 months, with each iteration having a duration of 4-6 months. Iterations will be completed in an overlapping fashion to reduce schedule length. Once an iteration enters the build activity, the analysis of the next iteration begins. The model is designed in this manner to prevent long acquisition times, which is important in preventing the acquisition of obsolete technology. In addition, the shorter time period reduces the traditional project costs of long multi-year development and acquisitions.

THIS PAGE INTENTIONALLY LEFT BLANK

# VI. MRD SYSTEM ARCHITECTURE

In conjunction with the IID process model a three-tier system's architecture was developed to provide the development team with a blueprint of the system. The system's architecture will evolve throughout development. The process model and system architecture go hand in hand. Each iteration of the process model has an associated system's architecture that evolves into the complete system at the same time iteration seven is complete. The systems architecture is broken into layers to create a three-tier architecture that includes a data, application, and presentation layer (Giachetti 2013).

Each iteration has a system architecture consisting of the components needed to deliver system functionality and capability. The system architecture for each iteration is broken down into detail in chapter six of this report. Each iteration builds upon the previous using software and hardware to deliver incremental capability. Upon completion of iteration seven, the final architecture presented in this report will be complete, shown in Figure 11.

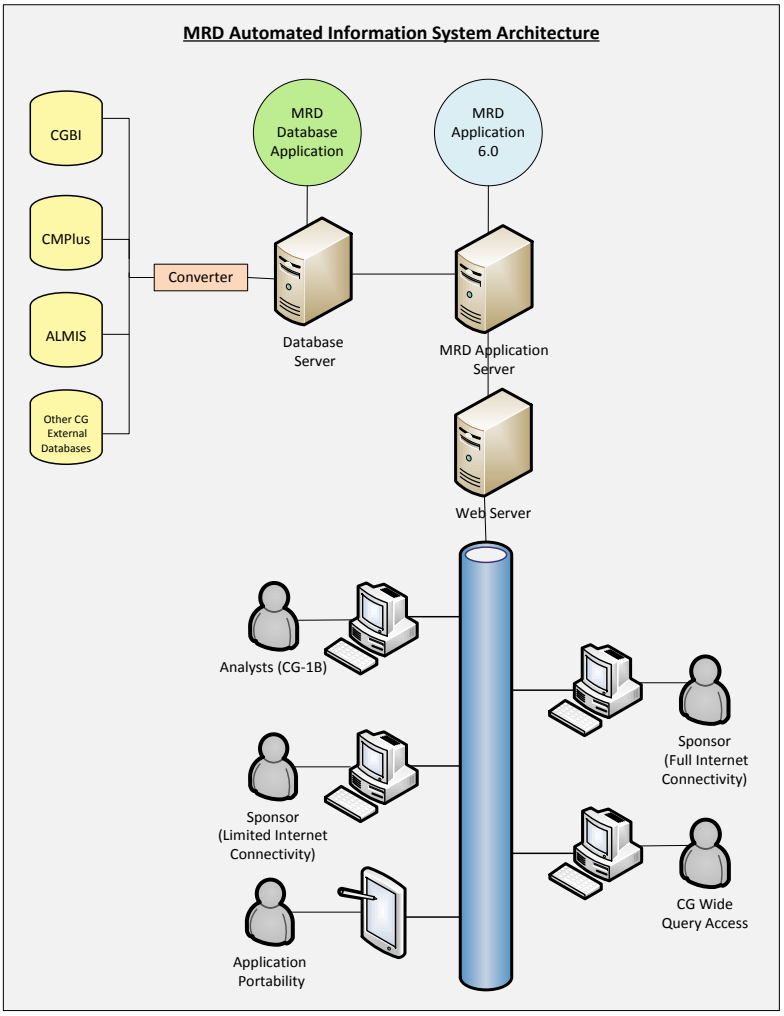


Figure 17. MRD System Final System Architecture

The system is built from a MRD database existing on a database server that provides a centralized location of storage for the systems data. The server will allow for simultaneous access of the system by multiple system users. Converters will be placed between the database server and external databases to allow for conversion of existing data in order to supplement the data within the MRD database. The priority databases for inclusion into the MRD system include Coast Guard Business Intelligence (CGBI), Asset Logistics Management Information System (ALMIS), and the Configuration Management Plus system (CMPlus). Other external databases that may need integration into the system are included in the architecture and can be determined during system development.

The MRD system will have a MRD Application that resides on a MRD Application server connected to a web server. This allows for delivery of the application to multiple users simultaneously across the CG network including analysts, sponsors with various levels of Internet connectivity, and users with general query access. The architecture shown in Figure 11 is the final architecture for the system.

With each iteration of the IID process model, the system architecture evolves. The major components, from the systems architecture, in relation to each iteration is shown in 11.

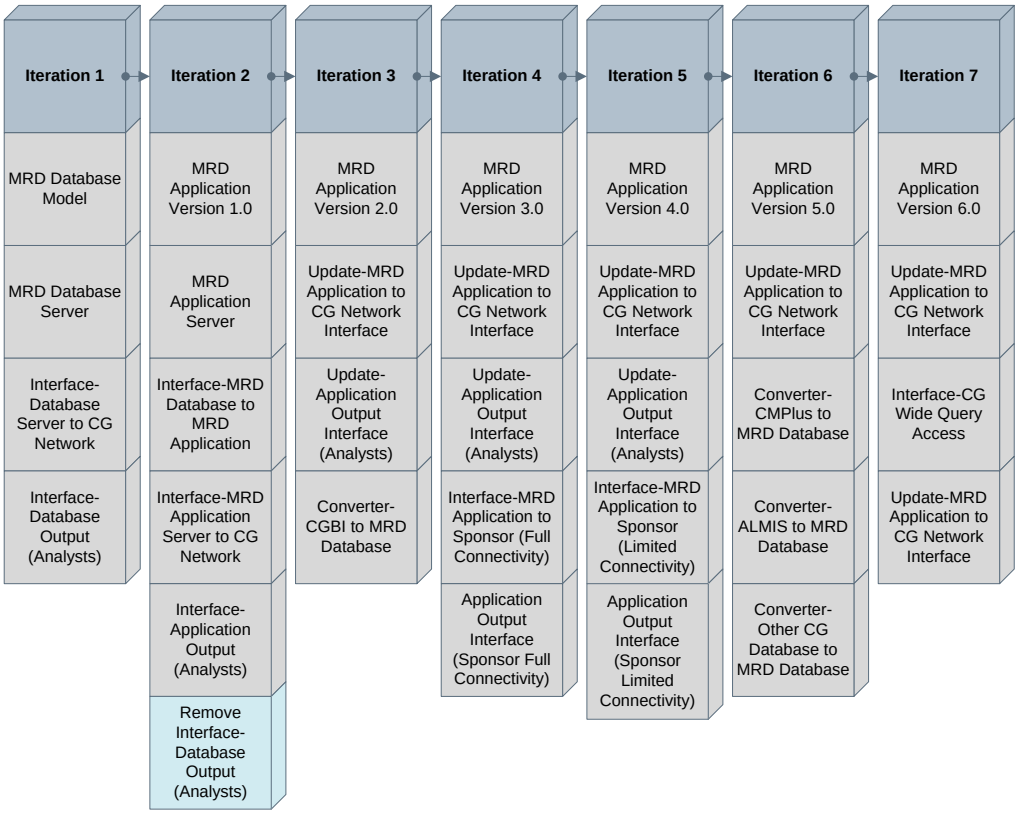


Figure 18. Process Model Component Mapping

The three-tier architecture evolves as the iterations are completed and is completed at the same time iteration seven is. The system's architecture is designed such that, if the CG decides at any point that they have need of additional capability, the architecture can accommodate it through the use of loose coupling (Giachetti 2013).

THIS PAGE INTENTIONALLY LEFT BLANK

# VII. PROCESS MODEL ITERATIONS

## A. ITERATION DETAILS & SYSTEM ARCHITECTURE

### 1. Overview

In additional to a logical buildable approach, the sequencing of the components within the model is based on selecting the capabilities with the highest return on investment being completed early on in development. Key components within the iteration architecture are the interfaces, especially those from system components to the users. Each iteration contains a set of interfaces to ensure the system is usable during the interim.

### 2. Iteration One

Iteration one contains the required system architecture to build the MRD database. The MRD Database, a software application residing on a database server, will be the main IT components of this iteration shown in Figure 19.

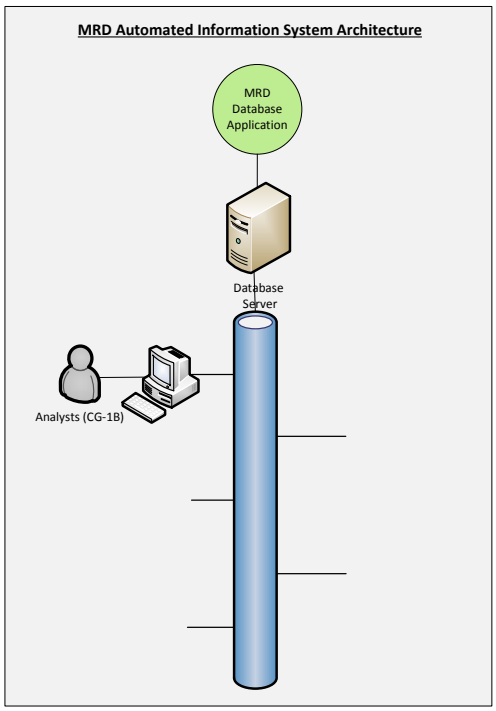


Figure 19. MRD System Process Model Iteration One System Architecture

While there are multiple databases within the CG, the MRD system will need a separate one containing only MRD specific data. As the system progresses through later iterations, access to additional CG databases will be addressed. The MRD database and server requires the development of an interface to connect to the CG network and enable data to flow across the network. With data capable of moving across the network for delivery onto hardware used by analysts, an interface will be developed to facilitate data download and use by analysts, Figure 19. This interface is a key component as it will be the first connection between the system and analysts and therefore the first realization of functionality and capability. The interface shall facilitate a user-friendly way of accessing data that is within the database for use until the next iterations architecture is realized and this interface is removed. Without this interface the capability provided through this iteration cannot be realized.

The database for the MRD system will only store data specific to MRDs. Large portions of the data needed to populate the system will be pre-loaded during the build phase. All MRD specific data cannot be realized up-front but as it is realized, the database can be added to through function 1.1.1 and 1.1.3. These functions will not only allow for the addition and update of data by the developer but will also provide data entry access for analysts. The analysts will also have the capability to delete data from the system that is no longer relevant, or may be provided when additional databases are integrated into the system during later iterations. A complete list of the functions and capabilities to be delivered in this iteration are listed in Table 5.

| Function Nbr | Function | Capability |
|---|---|---|
| | | Centralized Resource Management |
| 1.0 | Initiate MRA | x |
| 1.1 | Manage MRA Data | x |
| 1.1.1 | Input Data | x |
| 1.1.2 | Store Data | x |
| 1.1.3 | Update Data | x |
| 1.1.4 | Delete Data | x |

Table 5. Iteration One Function to Capability Mapping

### 3. Iteration Two

Iteration two will be the first phase of development where the application is introduced and integrated into the system, Figure 20.

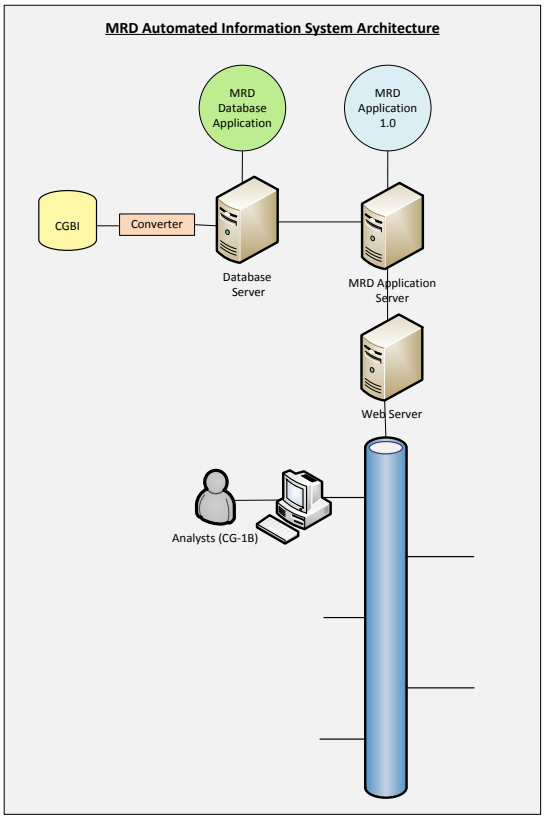


Figure 20. MRD System Process Model Iteration Two System Architecture

Development of iteration two's architecture will expand on iteration one's data collection capability by adding function 2.0, process MRA data. This will enable the analysts to begin to use of the basic functions needed to perform semi-automated manpower data collection and minimum analysis such as allowing the system to aid in identifying constraints (function 2.2.3) and identifying assumptions (2.2.4), Table 6. The application will allow for expansion of data collection by enabling the collection of data directly related to an individual MRD versus the general data collected and stored within the MRD database. This will include a means for analysts to enter the data found through historical data review, forecasting, interviewing, and observation. The application will be

capable of taking this data and populating the MRD templates and reports needed to continue through the MRD process. Analysts will be capable of recording work measurement characteristics and identifying the constraints and assumptions related to the individual MRD in process. The application delivered in this stage, version 1.0, is limited to delivering the functionality and capability shown in Table 6.

| Function Nbr | Function | Capabilities | |
|---|---|---|---|
| | | Centralized Resource Management | Centralized Manpower Data Analysis |
| 1.0 | Initiate MRA | x | x |
| 1.1 | Manage MRA Data | x | |
| 1.1.1 | Input Data | x | |
| 1.1.2 | Store Data | x | |
| 1.1.3 | Update Data | x | |
| 1.1.4 | Delete Data | x | |
| 1.2 | Create MRA | x | x |
| 1.2.1 | Create CG 5310 | x | |
| 1.2.2 | Determine MRA Type | | x |
| 1.2.3 | Categorize MRA | | x |
| 1.3 | Prioritize MRA | x | |
| 1.3.1 | Assign MRA Priority | | x |
| 1.3.2 | Integrate MRA into Master List | | x |
| 2.0 | Process MRA Data | x | x |
| 2.1 | Collect Data | x | x |
| 2.1.1 | Collect Work Matrix Data | x | |
| 2.1.2 | Collect Extant Data | x | |
| 2.1.3 | Collect Historical Data | x | |
| 2.1.4 | Collect Forecasting Data | x | |
| 2.2 | Prepare Data | x | x |
| 2.2.1 | Allocate Identifying Information | x | x |
| 2.2.2 | Record Work Item Measurements Characteristics | x | x |
| 2.2.3 | Identify Constraints | | x |
| 2.2.4 | Identify Assumptions | | x |

Table 6. Iteration Two Function to Capability Mapping

### 4. Iteration Three

Iteration continues to build the system by providing additional application capability, report generation, and the integration of data from the Coast Guard Business Intelligence (CGBI) system, Figure 21.

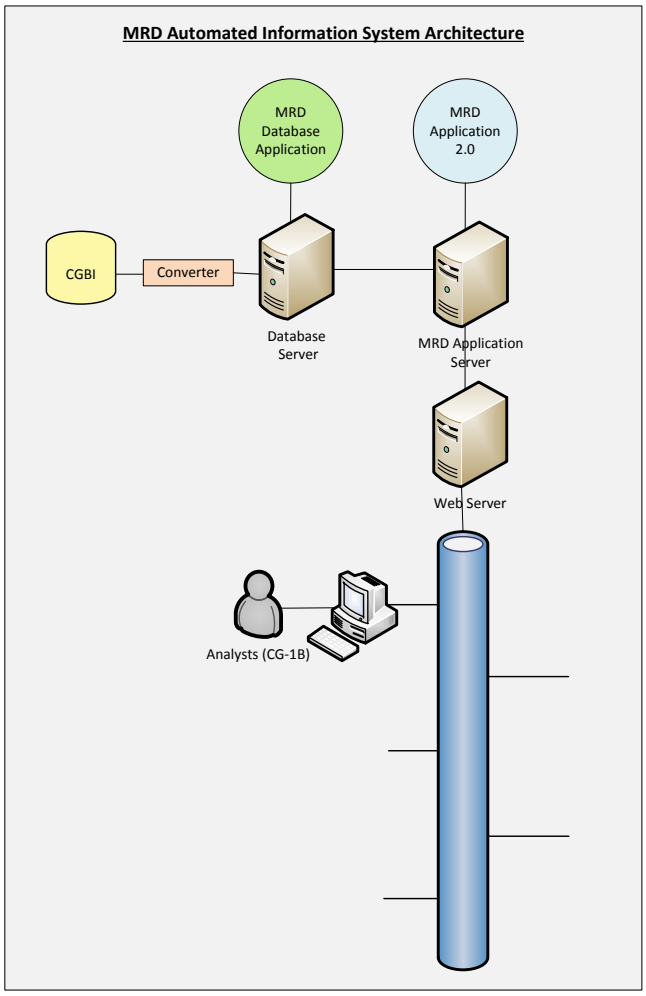


Figure 21. MRD System Process Model Iteration Three System Architecture

The CGBI system is an enterprise-wide system used by the CG to house data, produce reports, and perform analysis on numerous CG missions, tasks, people, supplies, training, logistics, and budget. The addition of an interface that allows the MRD database to access data within CGBI will make the MRD database more robust and allow some early indirect input from other CG organizational elements into manpower analysis.

This will also allow for data naturally collected through numerous other means and entered into CGBI, to easily flow into the MRD system without adding additional work on the part of analysts and Sponsor Units. Another large area of capability delivery in this iteration is through function 2.4, produce final data/work reports, and associated sub functions. This allows for the data and analysis conducted within function 2.0, process MRA data, to be realized. While new functionality is added during this iteration, functionality previously delivered may also be made more robust or improved based on the analysis activity completed in the early days of this iteration. The mapping of function to capability for iteration three is shown in Table 7.

| Function Nbr | Function | Capabilities | | |
|---|---|---|---|---|
| | | Centralized Resource Management | Centralized Manpower Data Analysis | Manpower Report Generation |
| 2.3 | Analyze Data | x | x | |
| 2.3.1 | Analyze OE Workload | | x | |
| 2.3.2 | Apply Assumptions | | x | |
| 2.3.3 | Apply Constraints | | x | |
| 2.3.4 | Identify & Update New Assumptions/Constraints | x | x | |
| 2.4.5 | Develop Possible OE Major Accomplishments | x | x | |
| 2.3.5 | Model OE Workload | | x | |
| 2.4 | Produce Final Data/Work Reports | | | x |
| 2.4.1 | Create Final Data/Work Report | | | x |
| 2.4.2 | Update WAC Presentation | | | x |
| 2.4.3 | Create conference Agenda | | | x |
| 2.4.4 | Create Work Matrix | | | x |
| 2.4.5 | Create MA List | | | x |
| 2.4.6 | Create Competency List | | | x |
| 2.4.7 | Perform Updates to Documents During WAC | | | x |
| 2.4.8 | Create Final Work Matrix | | | x |
| 2.4.9 | Create Final Report | | | x |
| 2.4.10 | Create WCA Report | | | x |

Table 7. Iteration Three Function To Capability Mapping

## 5. Iteration Four

Iteration four, like iteration three, adds more functionality to the MRD application but also adds an interface between the system and Sponsor Units, shown in Figure 22.

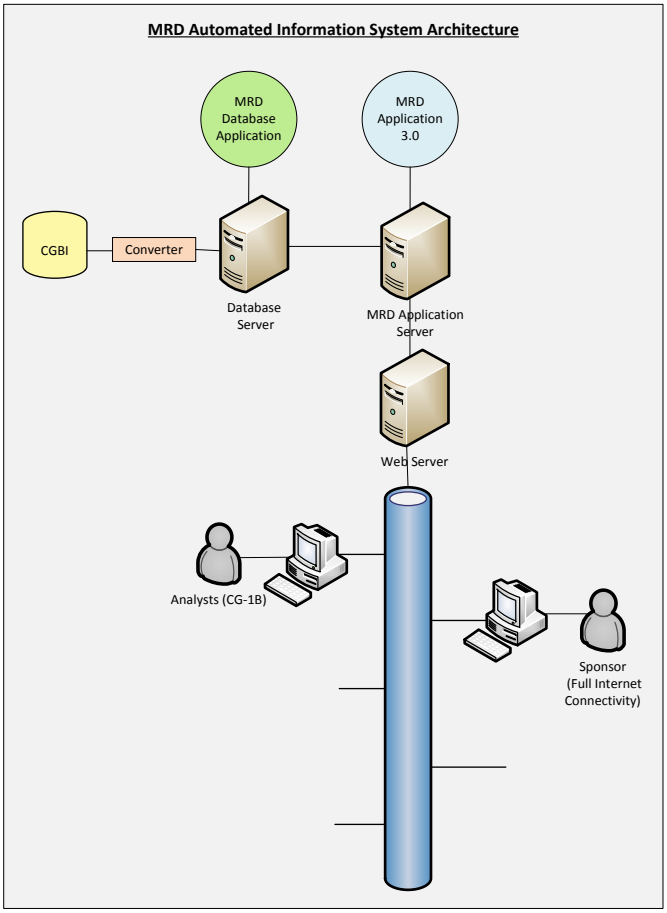


Figure 22. MRD System Process Model Iteration Four System Architecture

Iteration four adds function 3.0, Analyze Workforce Options, and all the sub functions to the system. Function 3.0 will include the development of the application to use the data and analysis form the system and diagram and model the data. The modeling will allow the final manpower determination to be based on scientific methodology. With all of function 3.0 being delivered, the analysts will have the capability to produce all of the documents for an MRD and print models and diagrams as needed. Within iteration four, function 2.3.6, Inport/Export MRA to Sponsor, will allow the movement of the MRA/MRD documents and data from the analysts to the sponsor. This function will only encompass the movement of the MRA/MRD document and not allow for more than editing of these documents. The next iteration will add functionality for sponsors to enter data directly. Iteration fours function to capability mapping is shown in Table 8.

| Function Nbr | Function | Capabilities | | | |
|---|---|---|---|---|---|
| | | Centralized Resource Management | Centralized Manpower Data Analysis | Manpower Report Generation | Manpower Resources/Data Access |
| 2.3.6 | Inport/Export MRA to Sponsor | x | | | x |
| 3 | Analyze Workforce Options | | x | | |
| 3.1 | Determine Force Mix Options | | x | | |
| 3.1.1 | Create Manpower Determinant Model (MDM) | | x | | |
| 3.1.2 | Identify Manpower Options | | x | | |
| 3.2 | Create MRA Options Report | | | x | |
| 3.3 | Analyze Viability of Options | | x | | |
| 3.4 | Produce Workload Data Diagrams | | x | x | |
| 3.4.1 | Create Workload by Work Type Diagram | | x | x | |
| 3.4.2 | Create Workload by Work Function Diagram | | x | x | |
| 3.4.3 | Create Worklaod by OE Sub-structure Diagram | | x | x | |
| 3.4.4 | Create Workload by Manpower Type Diagram | | x | x | |
| 3.4.5 | Create Workload by MA Diagram | | x | x | |
| 3.4.6 | Create Workload by Competency Diagram | | x | x | |
| 3.5 | Incorporate Sponsor Preferred Options | x | x | x | |
| 3.6 | Create Final MRD Report | x | x | x | |

Table 8. Iteration Four Function to Capability Mapping

This is the first stage of developing function 2.3.6 to allow use of the application by Sponsor Units. This iteration will see that functionality realized for those Sponsor Units with full Internet connectivity. Internet connectivity types are defined in detail in Chapter VII of this report. During this iteration, the application interface that the sponsor sees, to interact with the system, should be as robust as needed to maximize sponsor participation in conducting MRDs but does not need to have all the functionality that that analysts have. A study should be conducted to see what levels of interaction from the sponsor would be most beneficial to maximize sponsor participation at Sponsor Units with full Internet connectivity.

### 6. Iteration Five

Iteration five is focused on adding an interface for unit sponsors with various forms of limited Internet connectivity in addition to the functionality needed for Sponsor Units to conduct data input. The system architecture for iteration five is shown in Figure 23.

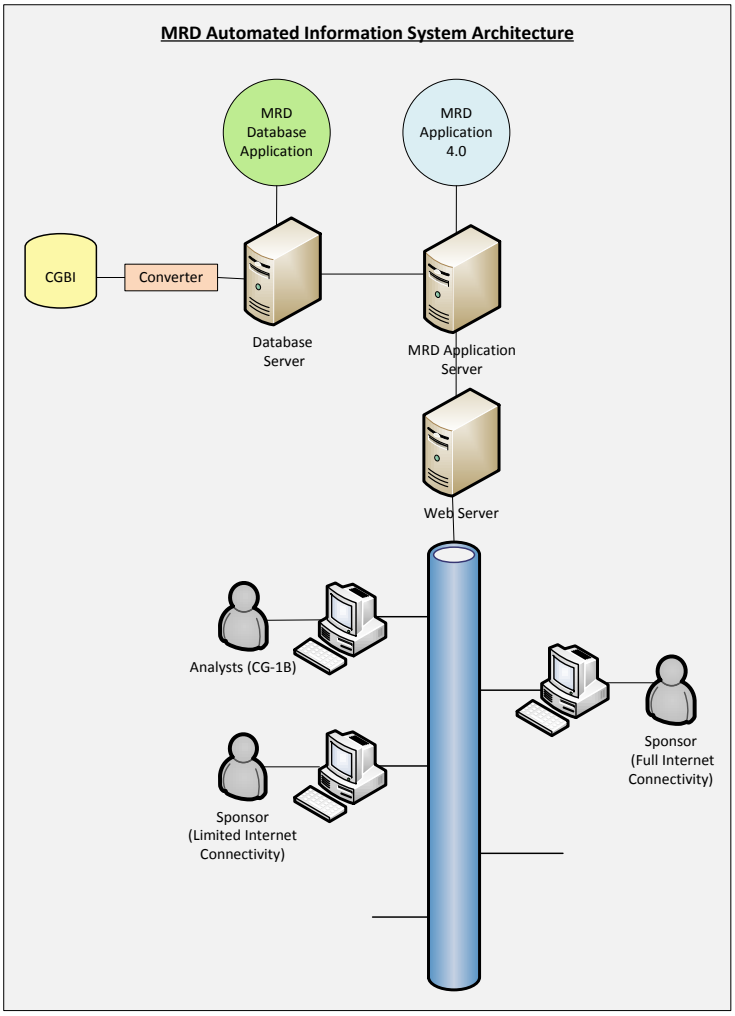


Figure 23. MRD System Process Model Iteration Five System Architecture

As discussed in Chapter VII, Sponsor Units with limited Internet connectivity do not have access to the Internet access with large bandwidth or fast speeds, such as that realized by land based units. Therefore development of version 4.0 of the application will include a 'lite' or mobile device type interface with only the functionality needed for a sponsor to provide the data and feedback necessary to assist CG-1B4 analysts. Keeping the application delivered to these unit sponsors limited will ensure the system functions on limited bandwidth and Internet speeds. A study, as completed in iteration four for Sponsor Units, should also be completed during the analysis phase of this iteration. This will ensure that the level of sponsor participation required is in line with what system functions can be delivered in a mobile style interface and available network parameters.

In addition, some units may not have any Internet accessibility during operations and deployments. These units require a means of working within the application without CG network access to include using the application, saving work completed, and upload of work and data upon connection to the Internet. It is recommended that an analysis of Internet connectivity be completed during the analysis activity of this iteration to ensure the interface and application access given to these units is optimal. The function to capability mapping for this iteration includes the addition of further capability and sub functions as shown in Table 9.

| Function Nbr | Function | Capabilities | | | |
|---|---|---|---|---|---|
| | | Centralized Resource Management | Centralized Manpower Data Analysis | Manpower Report Generation | Manpower Resources/Data Access |
| 1.0 | Initiate MRA | x | x | x | x |
| 1.1.1 | Input Data | x | | | x |
| 1.1.3 | Update Data | x | | | x |
| 2.1 | Collect Data | x | x | | x |
| 2.1.1 | Collect Work Matrix Data | x | | | x |
| 2.1.2 | Collect Extant Data | x | | | x |
| 2.1.3 | Collect Historical Data | x | | | x |
| 2.2.2 | Record Work Item Measurements Characteristics | x | x | | |
| 2.2.3 | Identify Constraints | | x | | |
| 2.2.4 | Identify Assumptions | | x | | |

Table 9. Iteration Five Function to Capability Mapping

## 7. Iteration Six

Iteration six makes the MRD database more complete than previous iterations by adding interfaces from additional CG databases in order to feed existing data into the MRD system, shown in Figure 24.

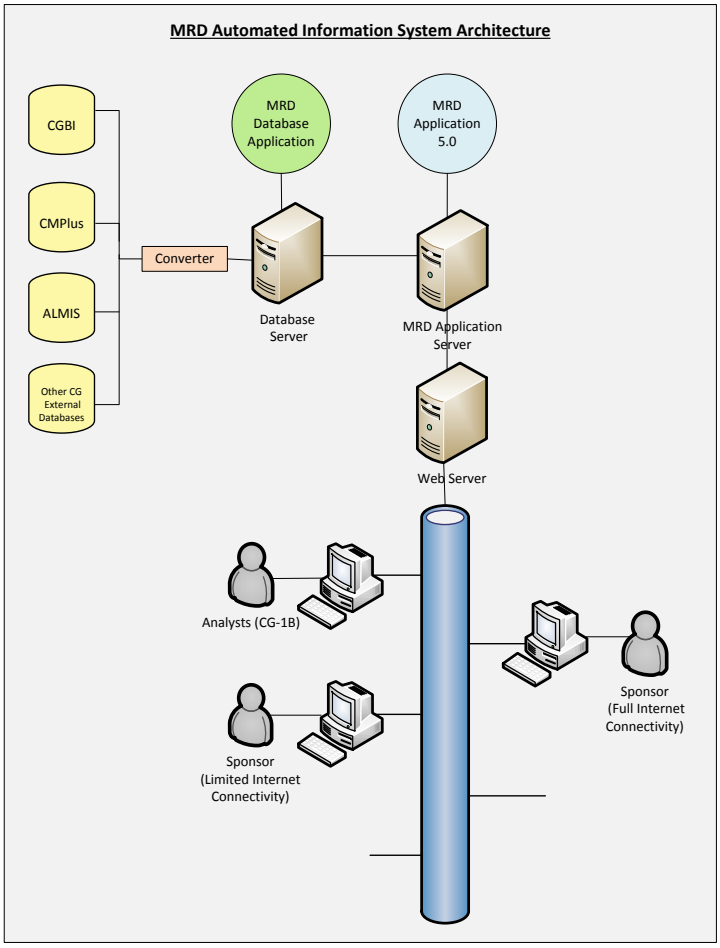


Figure 24. MRD System Process Model Iteration Six System Architecture

The two large main databases to have interfaces added are the Asset Logistics Management Information System (ALMIS) and the Configuration Management Plus system (CMPlus). In addition, other databases may be identified that need interfaces to feed data into the MRD database; this can be completed during this iteration and is represented in Figure 24 as "Other CG External Databases." The function to capability mapping for this iteration adds to the capability categories of earlier delivered functionality, this mapping is shown in Table 10.

| Function Nbr | Function | Capabilities | | | | |
|---|---|---|---|---|---|---|
| | | Centralized Resource Management | Centralized Manpower Data Analysis | Manpower Report Generation | Manpower Resources/Data Access | All CG Unit Type Data Access |
| 1.0 | Initiate MRA | x | x | x | x | x |
| 1.1 | Manage MRA Data | x | | | x | x |
| 1.2 | Create MRA | x | x | x | | |
| 1.3 | Prioritize MRA | x | | | | |
| 1.4 | Produce MRA Documents | x | | x | | |
| 2.0 | Process MRA Data | x | x | x | x | x |
| 2.1 | Collect Data | x | x | | x | x |
| 2.2 | Prepare Data | x | x | | x | |
| 2.3 | Analyze Data | | x | | x | |
| 2.4 | Produce Final Data/Work Reports | | x | x | | |
| 3.0 | Analyze Workforce Options | | x | x | x | |
| 3.1 | Determine Force Mix Options | | x | | | |
| 3.2 | Create MRA Options Report | | | x | | |
| 3.3 | Analyze Viability of Options | | x | | | |
| 3.4 | Produce Workload Data Diagrams | | | x | | |
| 3.5 | Incorporate Sponsor Preferred Options | | | | x | |
| 3.6 | Create Final MRD Report | | x | x | | |

Table 10. Iteration Six Function to Capability Mapping

## 8. Iteration Seven

Iteration seven is the final iteration for the process model covered in this report. Iteration seven delivers the final capability and functionality needed by analysts to realize all aspects of the system built herein. This iteration also adds two components; interface for CG wide query access and an interface for portable devices, shown in Figure 25.

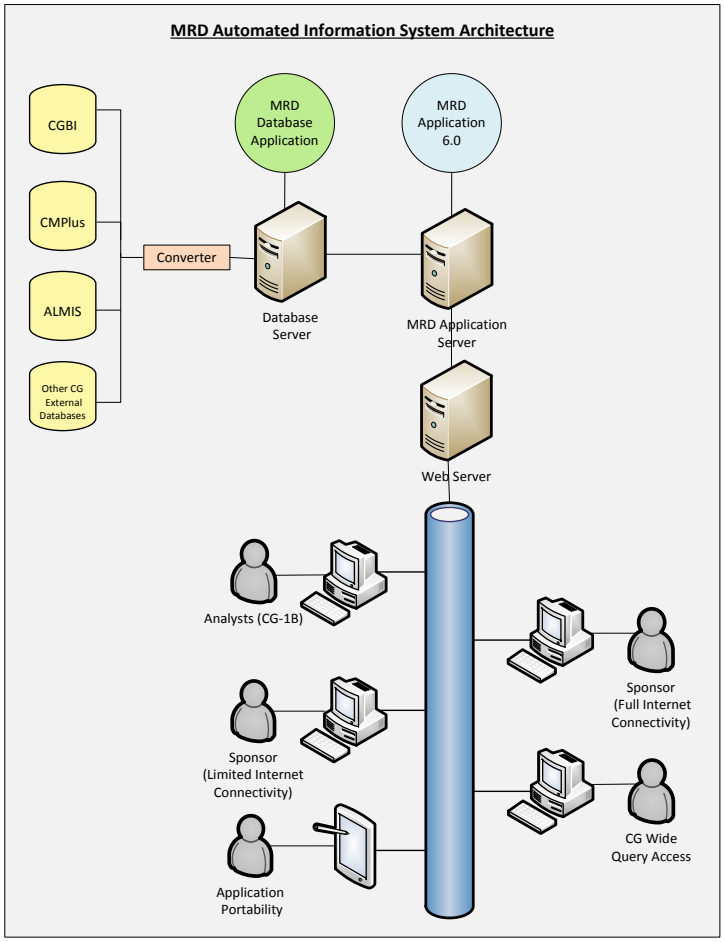


Figure 25. MRD System Process Model Final System Architecture

CG wide query access is the capability for any unit or person within the service to access the MRD system and obtain manpower raw data and completed MRDs. This will give the CG the capability needed to ensure that data used by personnel, outside of CG-1B4 and Sponsor Unit managers, is reliable and accurate. The biggest benefit will be seen by personnel managing acquisition projects, personnel managing CG human capital, and personnel providing data to organizations outside the CG such as DHS and Congress. The more accurate and reliable the data, the more capable the CG will be at obtaining and identifying the true personnel needs of the service for mission execution. Personnel within the CG will also be better able to translate personnel gaps and justify personnel increase requests. Lastly, the CG will have the ability to use existing manpower data when trying to forecast the personnel needs of newly acquired ships, aircraft, and boats.

The other addition to the architecture and capability within this iteration is an interface and application version for portable devices such as tablets; this will allow analysts to conduct field data collection without laptop and desktop hardware. The use of portable devices such as tablets has created an easier means of traveling and using hardware in the field. To maximize the portability and use of such systems, the application enhancements during this activity enable the software to deliver the 'lite' version of the software created in iteration five to tablet devices but with the functionality needed to conduct data collection. Analysts do not need to have full system capability when traveling to Sponsor Units, only the functionality needed to collect data during observation, interview, and Sponsor Unit meetings in addition to updating forms related to the Sponsor Unit. Modeling and analysis functions are not needed as that work can be completed when located back in the office at Headquarters. The last function to capability mapping for the system is shown in Table 11.

| Function Nbr | Function | Capabilities | | | | |
|---|---|---|---|---|---|---|
| | | Centralized Resource Management | Centralized Manpower Data Analysis | Manpower Report Generation | Manpower Resources/Data Access | All CG Unti Type Data Access |
| 1.0 | Initiate MRA | x | x | x | x | x |
| 1.1 | Manage MRA Data | x | | | x | x |
| 1.2 | Create MRA | x | x | x | | |
| 1.3 | Prioritize MRA | x | | | | |
| 1.4 | Produce MRA Documents | x | | x | | |
| 2.0 | Process MRA Data | x | x | x | x | x |
| 2.1 | Collect Data | x | x | | x | x |
| 2.2 | Prepare Data | x | x | | x | |
| 2.3 | Analyze Data | | x | | x | |
| 2.4 | Produce Final Data/Work Reports | | x | x | | |
| 3.0 | Analyze Workforce Options | | x | x | x | |
| 3.1 | Determine Force Mix Options | | x | | | |
| 3.2 | Create MRA Options Report | | | x | | |
| 3.3 | Analyze Viability of Options | | x | | | |
| 3.4 | Produce Workload Data Diagrams | | | x | | |
| 3.5 | Incorporate Sponsor Preferred Options | | | | x | |
| 3.6 | Create Final MRD Report | | x | x | | |

Table 11. Iteration Seven Function to Capability Mapping

## B. ITERATION SUMMARY

Upon completion of iteration seven, the process model presented in this thesis is complete as is the system architecture. As mentioned previously, additional iterations can be added if the CG sees the need for additional system functionality and capability. If additional iterations are incorporated they should follow the same guidelines presented within this report in regards to the activities within each iteration. To assist the development team, an operational overview of the system is presented in the following chapter.

THIS PAGE INTENTIONALLY LEFT BLANK

# VIII. OPERATIONAL OVERVIEW

Operationally, the system will create a seamless means of participation between Sponsor Units and analysts in order to complete MRDs. The CG network, through communication links, is the means by which data moves across the CG network, Figure 26.

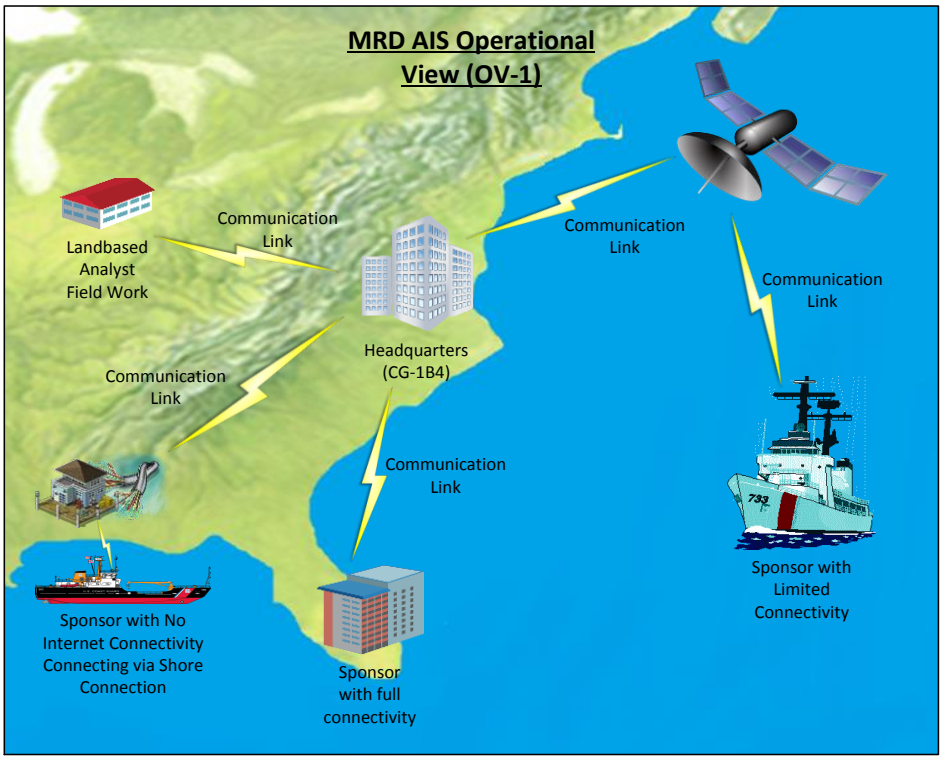


Figure 26. MRD System Operational View (OV-1)

Each sponsor unit is connected into the CG network differently. Some units have full time connectivity to the Internet while others do not. Sponsor Units complete multiple portions of the MRD and therefore need a means of communicated with CG-1B4 and linking into the MRD system. Sponsor Units can be classified as having one of four different types of Internet connectivity situations:

- Full Internet Connectivity Units: Sponsor Units with full time Internet connectivity. Usually units that are land based, INCONUS, or with wired Internet connectivity.

- Limited Internet Connectivity Units: Sponsor Units with less than full time Internet connectivity. This is usually a unit that is sea based such as CG cutters and aviation assets deployed on CG Cutters. These units have Internet connectivity, via satellite linkage, less than full time or at a limited bandwidth. Due to satellite linkage the Internet speed may not only be unavailable at times but also at speeds less than those seen from land based connected units.

- Non-Existent Connectivity: Sponsor Units with no Internet connectivity during operational active periods. Usually units that are smaller and sea based or air based. These units require the ability to use IT systems while deployed and the capability to have the data and work uploaded upon connection to land based Internet outlets.

- Portable Connectivity: For analysts conducting data collusion, or fieldwork, to include interviews, observation at Sponsor Units, and other data collection outside of CGHQ. This can be realized in many ways but at a minimum should include the ability of analysts to complete data collection via portable device and have it uploaded to the system via some version of connectivity.

With the CG being spread across the United States and overseas it is vital that the system is capable of delivering the required system functionality for each user type. By connecting all levels of users, the entire CG can participate in resource management and benefit from different aspects of the systems capability.

# IX. CONCLUSION

## A. OVERVIEW

The MRD system is a small enterprise-wide system that, after development, will provide the CG with the capability to accomplish MRDs, from start to finish, with limited input from analysts. The automated system will provide centralized resource management, centralized manpower data analysis, manpower report generation, manpower resource/data access, and all CG unit query access. To ensure the capabilities are delivered this thesis argues and presents the IID process model as the best model option for development of the system. In addition, the IID process model will not work without a three-tier architecture, which is presented herein. Lastly, up-front engineering analysis completes the necessary tools for system development.

To understand the problem faced by the CG a complete up-front analysis of the stakeholder needs was completed. The up-front analysis included an evaluation of current CG-1B4 business practices to understand both the business processes involved and the current tools used to assist analysts. The business processes involves both the MRA and the MRD; the MRA produces the manpower needs and the MRD results in how the needs are met. Both of these processes will be automated and result in one system called the MRD system. Current automation and tools of completion are limited to Microsoft Office© products. The lack of an IT system, with associated software, for the MRD process limits the CG to producing MRDs that are not reliable, repeatable, or defendable. The MRD system will consist of tools and components needed to ensure data, analysis, and modeling are based on scientific formulas, reliable and consistent data, and produce models and reports to provide better snapshots of human capital needs.

To choose the best model for the MRD system, four process models were considered, which included the waterfall, "Vee," spiral, and IID. The analysis found that both the waterfall and "Vee" model provided a robust and extensive development process with all of the capabilities required delivered at project completion. These two models also require stable up-front systems engineering analysis including the requirements.

There was little room for feedback and changes, which increased the risk of using these models; therefore these models were not selected. The spiral model is slightly different than the waterfall and "Vee," containing four phases of planning, risk analysis, engineering, and evaluation. This allows for some capability to be delivered throughout development but uses a lot of development time for risk analysis. The MRD system, while containing risk, does not contain the same risk as for example a ten-year billion dollar multi-ship development and acquisition. The spiral model was not a good model choice for a smaller project. The IID process model analysis revealed a process that accommodates up-front engineering analysis but is designed with feedback paths to accommodate changes without altering the success of the project. With multiple increments of development, the CG would have a project that is divided into smaller, more management, less complex, and less risky increments. The IID model's feedback paths also allow the user to be involved in development. The user being involved in this project is vital. The MRD system needs has functionality and capability directly related to completing the steps involved in a MRD for analysts. Lastly, the IID model provides from the completion of the first iteration onto the last, when final functionality and capability are delivered.

The IID process model designed for the CG's MRD system will include seven iterations with five associated activities within each; analysis, build, implement, deploy, and support. Feedback lines can be seen in Figure 16 from the build and implement activity leading into the analysis activity of the next iteration. Additional feedback lines were not drawn in but any other feedback received after implement, deploy, or support can be incorporated during the subsequent iteration's analysis activity. The IID process model, shown in Figure 15, displays the 18-month development schedule. Each iteration will progress through the five activities over the course of 4-6 months with analysis taking more than the other activities. The iterations occur in an overlapping fashion to ensure the project's overall schedule remains short to reduce cost and schedule.

Each iteration delivers capability, which gives the CG improved workflow and MRD analysis throughout the project development. The seven iterations, allow the development team to address the building of the system which allows the team to have a

better understanding of the project scope, increase their confidence in developing the system, and act upon feedback gained through development to ensure the system meets user needs (Spence and Bittner 2005). The seven iterations will also aid the CG in preventing the numerous reasons projects fail by limiting the technical complexity, providing up-front high level systems engineering work, allowing the development of lower level requirements within each iteration, providing numerous implementation periods that incorporate test and evaluation, and allow for user participation in development. This will also allow the project development team to assess the risk of each iteration during the analysis activity and ensure it is reduced and mitigated appropriately.

This thesis also provides a three-tier architecture that will be used a system blueprint, shown in Figure 11. The architecture consists of components that build three layers of the architecture including a data, application and presentation layer. The data layer provides the components and means to support a database with included software and converters for integration with current CG databases. The application layer provides the components and means to deliver the MRD software application, which provides all of the analysis and modeling operations. The final layer is the presentation layer which enables delivery of the both the database and application functions and capability to the user. The three-tier architecture for the MRD system will evolve in a consistent progression with the IID process model, which can be seen in Figures 19 through 25. This MRD system architecture provides the CG with a target system architecture that meets CG policies and requirements for development. The architecture also provides the development team with a visualization of the system for better understanding of the system and interoperability of components.

This thesis presents the bests mean, through up-front systems engineering analysis, a MRD IID process model, and three-tier systems architecture, to develop the MRD system. This work will allow the development team to have a thorough understanding of the needs and capabilities the system is required to have using written documentation and modeling. In addition, with the tools provided herein will allow the development team to derive a better list of requirements for contract specifications

through better system understanding. With a successful development of the MRD system, the CG will have greatly improved their ability to manage human capital by providing analysts the required functionality and capability to meet the needed completion rate of 128 every five-year cycle. The IID process model developed for this system, will give the CG a low risk, reduced cost, and quicker means of acquiring the system than previous attempts. With CG manpower costs being approximately 70% of the CG's overall costs on a yearly recurring basis decisions need to be based on reliable accurate data (USCG 2005).

## B. CG POLICY

The development of the IID process model and three-tier architecture enables the CG to meet the policies governing the development of IT systems. The two major policy manuals governing IT system development are the Command, Control, Communications, Computers and Information Technology (C4&IT) Enterprise Architecture Policy Manual and the C4&IT System Development Life Cycle Policy Manual. Compliance with these manuals is necessary to proceed through development and acquisition. Therefore, to maximize benefit to the CG, the work completed in this report, including the IID process model, system architecture, and written work contained in Chapter I, was completed in line with these manuals.

The CGs C4&IT Enterprise Architecture Policy Manual requires that all projects demonstrate three major project components: as-is, transition, and target (USCG 2004) as shown in Figure 27.

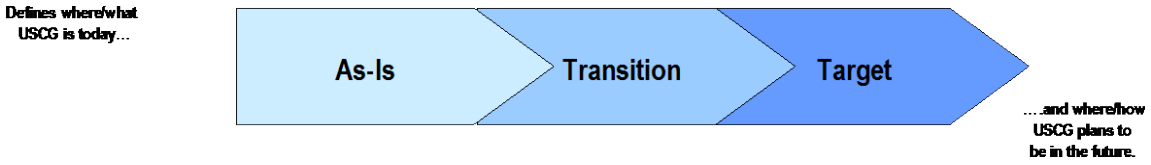


Figure 27. Coast Guard Architecture Key Components (USCG 2004)

The as-is component is covered by the description of current business practices within CG-1B4, written out in Chapter I, Section B. The transition component is

demonstrated through each iteration by way of both the system architecture and associated function to capability mapping by ensuring the project and resulting system continue to progress toward an improvement over current methods in use to complete MRDs. Lastly, the target component is demonstrated in iteration sevens final system architecture and final delivery of function and capability mapping.

The second requirement within the C4&IT Enterprise Architecture (EA) Policy Manual is the use of the DoDs Architectural Framework (DoDAF). The IID process model, MRD system architecture, systems engineering work, and operational concept graphics contained within this report meet this requirement. Specifically, the models and graphics contained herein fit into the following DoDAF views:

- Capability Vision (CV-1): a graphical representation of the MRD system and the capabilities associated with the system, shown in Figure 9.
- Capability Phasing (CV-3): a graphical representation of the MRD system phases and the related functions and capabilities, shown in Table 5-11.
- Operational View (OV-1): a high level graphical representation of the MRD system in respect to the organizational elements and the connection and communication between each element, shown in Figure 26.
- Project Timelines (PV-2): a timeline in relation to the MRD system project milestones, shown in Figure 14 and 15.
- Systems Interface Description (SV-1): a model displaying the components the MRD system will consist of and the components that connect and integrate the system, shown in Figure 17 and Figures 19-25.
- Systems Functionality Description (SV-4): a graphical representation of MRD system functions, shown in Figures 10-13.

The second policy manual reviewed for system development was the C4&IT System Development Life Cycle (SDLC) policy. This policy manual requires each project to develop a SDLC tailoring plan, which includes the following:

- Products to be produced
- Events to be conducted
- Description of the new system

- A funding strategy
- A development approach
- Size of the system
- Cost of the system
- Complexity of the system
- Interfaces and interdependency of the system
- Organizational impact the system will have

While all these components are not covered, such as funding strategy and cost of the system, the other components have been covered throughout the report and will enable the CG to better estimate the cost of the system and develop a funding strategy to meet the process models development schedule.

## C. WAY FORWARD

As previously discussed, the IID process model is very flexible and can accommodate the acquisition documentation, requirements, and events needing to be accomplished, fitting them into the activities of each iteration that are most appropriate. An acquisition representative will now need to analyze the model iteration and place the acquisition decision events and milestones within. The system architecture presented herein is also flexible. If other required or desired components are realized when reviewing this report, the CG can easily be add or connect IT components to the system architecture. The development team should use caution when adding components to ensure the additions meet functional or non-functional requirements as presented herein. The CG can take the tools provided herein and immediately begin on system development.

# LIST OF REFERENCES


Admin. 2009 “Disadvantage of V-Model.” Accessed February 26 2014. http://embedsoftdev.com/tag/disadvantage-of-v-model/.

Bittner, Kurt. 2006. “Driving Iterative Development with Use Cases.” IMB Developer Works Technical Library, March 14. http://www.ibm.com/developerworks/rational/library/4029.html.

Blanchard, Benjamin S., and Fabrycky, W. Wolter J. 2010. *Systems Engineering and Analysis*. Upper Saddle River, NJ: Prentice-Hall.

Block, Michael, Blumberg, Sven, and Laartz, Jurgen. 2012. “Delivering Large-Scale IT Projects On Time, On Budget, and On Value.” Accessed April 2014. http://www.mckinsey.com/insights/business_technology/delivering_large-scale_it_projects_on_time_on_budget_and_on_value.

Boehm, Barry. 2014. “The Spiral Model.” Accessed June 3, 2014. http://en.wikipedia.org/wiki/Spiral_model.

Bredemeyer Consulting. 2014.“Motivating Software Architecture.” Accessed April 14. http://www.bredemeyer.com/why.htm.

Brooks, Frederick. P. Jr. 1987. “No Silver Bullet: Essence and Accidents of Software Engineering.” *IEEE Computer* 20: 10-19.

California Department of Transportation. 2007. “System Engineering Guidebook for ITS.” www.fhwa.dot.gov/cadiv/segb/files/segbversion2.pdf.

Cauwenberghe, Pascal. Van. 2002. “Going Round and Round and Getting Nowhere Extremely fast? Another Look at Incremental and Iterative Development.” *Methods & Tools* 10: 13-20. http://www.methodsandtools.com/PDF/dmt0402.pdf.

CG-1B4. 2008. *MRD AIS Development and Support Plan, Version 2.3*. Washington DC.

Chodhury, Abdullah Al Murad and Arefeen, Shamsul. 2011 “Software Risk Management: Importance and Practices.” *International Journal of Computer and Information Technology* 02: 49–54.

Congress. 2013. H. R. 1232 (Introduced-in-House). Accessed May 06, 2014. http://docs.house.gov/billsthisweek/20140224/BILLS-113hr1232-SUS.xml#toc-H2BBAE1FDF1444D26AF48FB1961B7B2E9.

Creel, Rita. and Ellison, Robert.J. 2007. “Acquisition Overview: The Challenges.” *Software Engineering Institute.* Accessed June 04, 2007. http://resources.sei.cmu.edu/asset_files/WhitePaper/2004_019_001_293550.pdf.

Library of Congress. “Federal Information Technology Acquisition Reform Act (2013-2014)”. Accessed March 2014. http://beta.congress.gov/bill/113th-congress/house-bill/1232.

Debono, Mark. “Functional vs. Nonfunctional Requirements.” Accessed June 2014. http://reqtest.com/testing-blog/functional-vs-non-functional-testing/.

Durkovic, Ozren and Rakovic, Lazar. 2009 “Risks in Information Systems Development Projects.” *Management Information Systems* 4: 13-19. http://www.ef.uns.ac.rs/mis/archive-pdf/2009 - No1/MIS2009_1_3.pdf.

Eggen, Dan and Witte, Griff. 2006. “The FBI’s Upgrade That Wasn't.” *Washington Post*, August 18. http://cs.gmu.edu/~mlocasto/research/securehealth/content/post-VCF.pdf.

Fairley, Richard. E. and Willshire, Mary. Jane. 2005. “Iterative Rework: The Good, the bad, and the Ugly.” *Computer* 38: 34–41.

Federal Reserve. 2011. “Supervisory Guidance on Model Risk Management.” *Federal Reserve*. Accessed April 2014. http://www.federalreserve.gov/bankinforeg/srletters/sr1107a1.pdf

Gallagher, Brian. 2008. “Identifying Acquisition Patterns of Failure Using System Archetypes: Finding the Root Causes of Acquisition Problems.” Software Engineering Institute. www.sei.cmu.edu/library/assets/gallagher-gsaw08.pdf.

Government Accountability Office (GAO). 2008. “Information Technology: OMB and Agencies Need to Improve Planning, Management, and Oversight of Projects Totaling Billions of Dollars.” Washington D.C. http://www.gao.gov/products/GAO-08-925.

Government Accountability Office (GAO). 2010. “Department of Homeland Security: Assessments of Selected Complex Acquisitions.” Washington D.C. Accessed June 18, 2014. http://www.gao.gov/products/GAO-10-588SP.

Giachetti, Ronald. 2010. *Design of Enterprise Systems: Theory, Architecture, and Methods*. Boca Raton: CRC Press.

Giachetti, Ronald. 2010. Design of Enterprise Systems Course. Monterey, CA.

Giachetti, Ronald 2013. Enterprise Architecture Course. Monterey, CA.

Howerton, Joseph. 2009. “FBI Virtual Case File project (a case study).” *Slideshare*.net. http://www.slideshare.net/JosephHowerton/is-430-fbivcf.

Hurricane Softwares. “Understanding the pros and cons of the Waterfall Model of software development.” *Tech Blog; for developers, designers, & SEO Specialists.*

Accessed February 26, 2014. http://www.hurricanesoftwares.com/understanding-the-pros-and-cons-of-the-waterfall-model-of-software-development/.

INCOSE. 2004. "What is System Engineering?" Last updated June 14. http://www.incose.org/practice/whatissystemseng.aspx

Jones, Capers. 1996. "Strategies for Managing Requirements Creep." *Computer* 29: 92–94. http://www.students.science.uu.nl/~3092062/papers/11.PDF.

Jones, James. 2011. "Software Acquisition: Reducing Risks." Defense AT&L. Nov-Dec. Accessed May 2014. http://www.dau.mil/pubscats/ATL%20Docs/nov_dec11/Jones.pdf.

Larman, Craig. 2004. *Agile & Iterative Development: A Manager's Guide*. Boston: Addison-Wesley.

MacCormack, Alan. 2001. "Product-Development Practices that Work: How Internet Companies Build Software." MIT Sloan Management Review 42: 75–84.

McDonnel, Steve. 2004. *Code Complete*. Redmond: Microsoft Press.

Melonfire. "Understanding the pros and cons of the Waterfall Model of software development," *TechRepublic*. Accessed April 15, 2014. http://www.techrepublic.com/article/understanding-the-pros-and-cons-of-the-waterfall-model-of-software-development/.

Munassar, Nabil Mohammed Ali and Govardhan A. "A Comparison Between Five Models of Software Engineering." *International Journal of Computer Science* 7, no 5 (2010). 94–101. www.IJCSI.org.

Nieslen, Jakob. "First Rule of Usability? Don't Listen to Users." *Nielsen Norman Group*. August 5, 2001. http://www.nngroup.com/articles/first-rule-of-usability-dont-listen-to-users/.

Nieslen, Jakob. 2012 "Usability 101: Introduction to Usability." *Nielsen Norman Group*. Last updated January 4. http://www.nngroup.com/articles/usability-101-introduction-to-usability/.

Oxford English Dictionary. 2014. "Software." *OED Online*. Accessed April 03. http://www.oed.com/view/Enry/183938?redirectedFrom=software#eid.

Office of Inspector General (OIG). 2005. "Statement of Glenn A. Fine Inspector General, U.S. Department of Justice before the Senate Committee on Appropriations Subcommittee on Commerce, Justice, State and the Judiciary concerning The Federal Bureau of Investigation's Trilogy Information Technology." Washington D.C.: Office of Inspector General. http://www.justice.gov/oig/testimony/0502/.

Office of Inspector General (OIG. 2005. "The Federal Bureau of Investigation's Management of the Trilogy Information Technology Modernization Project." Washington D.C.: Office of Inspector General. https://www.justice.gove/oig/reports/FBI/a0507/final.pdf.

Pan, Jiantao. 1999. "Software Testing." *Carnegie Mellon University.* Spring. http://www.ece.cmu.edu/~koopman/des_s99/sw_testing/.

Poulsen, Kevin. 2004. "Software Bug Contributed to Blackout." *Security Focus News*. February 11. Accessed June 18, 2014. www.securityfocusnews.com/news/8016.

Reilly, Sean. 2012. "DoD IT Projects Over Budget, Behind Schedule." *Defense News.* July 23. Accessed June 18, 2014. http://www.defensenews.com/article/20120723/DEFREG02/307230015/DoD-Projects-Over-Budget-Behind-Schedule.

Rowley, David. 1996. *Field Methods Casebook for Software Design*. New York: John Wiley & Sons, Inc.

Royce, Dr. Winston. W. 1970. *Managing the Development of Large Software Systems*. Paper presented by Dina Zeliger at the Institute of Electrical and Electronics Engineers WESCON. Los Angeles, CA, August. www.cs.huji.ac.il/~feit/sem/se09/Waterfall.pdf.

Schafrik, Franki. 2011. "A Practical Guide to Developing Enterprise Architecture." *IMB Developer Works.* October 18. Accessed June 3, 2014. http://www.ibm.com/developerworks/rational/library/enterprise-architecture-maximum-value/.

Scheinholtz, Michael. 1998. "System Architecture Approaches." *Carnegie Mellon University*. Spring. http://users.ece.cmu.edu/~koopman/des_s99/system_architecture/

Sotirovski, Drasko. 2001. "Heuristics for Iterative Software Development." *IEEE Software 19:* 66–73.

Spence, Ivar and Bittner, Kurt. 2005. "What is iterative development?" *IBM Developer Works.* Accessed April 2014. http://www.ibm.com/developerworks/rational/library/may05/bittner-spence/bittner-spence-pdf.pdf.

Standish Group. 2009. "CHAOS Summary 2009: The 10 Laws of CHAOS." Accessed May 5, 2014. http://emphasysbrokeroffice.com/files/2013/04/Standish-Group-CHAOS-Summary-2009.pdf.

Standish Group. 2014. "The Standish Group Report: Chaos." Accessed May 5. www.projectsmart.co.uk/docs/chaos-report.pdf.

Thibodeau, Patrick. 2002 “Study: Buggy Software Costs Users, Vendors Nearly $60B Annually.” *Computer World*, June 25. http://www.computerworld.com/s/article/72245/Study_Buggy_software_costs_users_vendors_nearly_60B_annually.

U.S. Coast Guard. 2005. *Coast Guard Strategic Cost Manual, COMDTINST M7000.4.* Washington, DC.

U.S. Coast Guard. (n.d.*). Coast Guard Staffing Logic and Manpower Requirements Manual, Vol II (COMDTINST M5310.5)*, Washington, D.C.

U.S. Coast Guard. 2006. *Commandant’s Intent Action Order #8 - Human Resource Strategies To Support Coast Guard Maritime Strategy*. Washington D.C.

U.S. Coast Guard. 2008. “Manpower Requirements Determination (MRD) Enterprise Development Team Charter (Update 1). http://www.uscg.mil/ff21/docs/MRD_team_charter_updated.pdf.

U.S. Coast Guard. 2008. “MRD AIS Functional Requirements.” Washington, D.C.

U.S. Department of Transportation. 2007. *System Engineering for Intelligent Transportation Systems. Accessed June 3, 2014.www*.ops.fhwa.dot.gove/publications/seitsguide/seguide.pdf.

Wang, Qiushi and Yang, Zhao. “A Method of Selecting Appropriate Software Architecture Styles: Quality Attributes and Analytic Hierarchy Process.” University of Gothenburg, 2012.

Whitcomb, Clifford. n.d. “An Architecture-Based Systems Engineering Methodology.” Monterey, CA. https://cle.nps.edu/access/content/group/30aaca71-0084-4a82-9c7e-4690f4fa537a/Modules Sp10/Module1-IntroArchitecture/An Architecture-BasedSEMethodology-WhitcombNotes.pdf.

*Wikipedia*. 2014a. “File: Agile Project management by Planbox.png.” Accessed March 27. http://commons.wikimedia.org/wiki/File:Agile_Project_Management_by_Planbox.png.

*Wikipedia*. 2014b. “Northeast Blackout of 2003.” Accessed April 13. http://en.wikipedia.org/wiki/Northeast_blackout_of_2003.

*Wikipedi*a. 2014c. “Windows NT 3.1.” Accessed April 3. http://en.wikipedia.org/wiki/Windows_NT_3.1 (accessed April 3, 2014).

THIS PAGE INTENTIONALLY LEFT BLANK

# INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California